L.2.1987-459

Senior High School

TEACHER RESOURCE MANUAL

Computer Processing 10-20-30

1987

Curriculum

Alberta
EDUCATION

DDN
416999

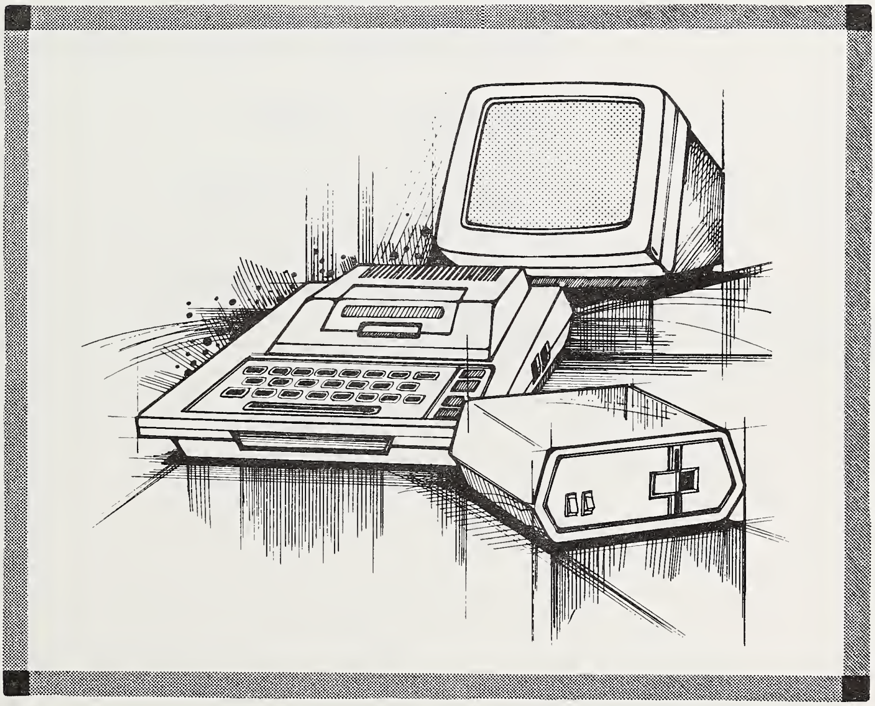
Digitized by the Internet Archive
in 2016

https://archive.org/details/computerprocessi1012albe

CANADIA
C2
AUG - 6

COMPUTER PROCESSING 10–20–30
TEACHER'S MANUAL

1987



Alberta
EDUCATION

ALBERTA EDUCATION CATALOGUING IN PUBLICATION DATA

Alberta. Curriculum Branch.
Computer processing : teacher's manual.

ISBN 0-920794-55-6

1. Electronic data processing -- Study and teaching (Secondary) -- Alberta.
2. Computers -- Study and teaching (Secondary) -- Alberta. I. Title.

QA76.27.A333 1987 001.64

ACKNOWLEDGEMENTS

This manual has been developed by Mel Fisher, University of Lethbridge, and Ray Lautt, M.E. LaZerte Composite High School, Edmonton. The suggestions that the authors give in this manual are the result of many years of successful experience in teaching computer-related courses. We appreciate their generous sharing of expertise and resource materials.

Preparation of this manual was authorized by A.A. (Scotty) Day, and Garry Popowich, Associate Directors, Curriculum Branch, Alberta Education, and co-ordinated by Sharon Prather, Project Manager, Senior High School Programs.

Production support was provided by Curriculum Branch staff:

Editing	– Anita Jenkins – Helen Scarth
Copy Editing	– Lisa McCardle – Kim Blevins
Word Processing	– Jan Cruickshank Lin Gray Denise Hugman Esther Yong

PREFACE

The purpose of this manual is to help teachers meet the program objectives for the revised Computer Processing 10-20-30 curriculum. This manual:

a) clarifies terms used and expectations outlined within the "Topics" and "Learning Tasks" sections of each module included in the curriculum

b) helps with long- and short-range planning by providing suggestions on how to select the modules, and/or how to sequence the topics within those modules

c) suggests practices which will encourage effective and efficient use of the learning resources and laboratory facilities

d) recommends how student progress can be evaluated

e) suggests alternative teaching strategies.

This is a support document, which is not mandatory in any way. It suggests methods that are based on practices used by successful and experienced educators in Alberta. Since no document of this type can address every situation, only suggestions that are appropriate should be used.

This manual is unbound, and punched for three-ring binders. This format will allow teachers to add personal references and/or notes, or move the "Teaching Suggestions" section next to the specific module from the curriculum guide. In addition, this format will facilitate updating the content as necessary.

TABLE OF CONTENTS

Page

Acknowledgements iii

Preface iv

Program Objectives 1

Flowchart of Modules 2

Planning

Long-Range Planning 5
Short-Range Planning 8

Sample Lesson Plans 9

Teaching Suggestions

Module 1: Computer Information Systems 23
2: Overview of Software 27
3: Applications: Data Entry 31
4: Applications: Word Processing 35
5: Applications: Computer Simulations 39
6: Applications: Data Base 43
7: Applications: Electronic Spread Sheet 47
8: Overview of Introductory Programming Language 51
9: Fundamentals of Input/Output 57
10: Introduction to Advanced Computer Programming Techniques 63
11: Advanced Computer Programming Techniques 67
12: Extended Programming Project 71
13: Introduction to Second Programming Language 75
14: Applications in Second Programming Language 79
15: Extended Project in Second Programming Language 81
16: Graphics 83
17: Systems Analysis and Program Development 85
18: Machine/Assembly Language 89

Evaluation 93

Resources 99

Appendices

Appendix A: Sample DOS Exercise for an Apple Lab 107
B: Professional Documentation Standards for Programs 108
C: Computer System Relationships 109
D: Sample Programming Project at Two Levels of Difficulty 110
E: Computing Language Primitives as Implemented in BASIC and Pascal 111
F: Suggested Evaluation of Computer Programming Assignments 112
F-1: Apple II Memory Organization and Usage 113
F-2: CBM 8032 Memory Organization and Usage 114
F-3: Memory Architecture 115
G: Text and Graphic Format Sheet 116
H: Comparison of Pseudocode, Structure Diagram and Flowchart for Statement of Algorithms 117

Correlation of Course Content with Learning Resources 118

PROGRAM OBJECTIVES

COMPUTER PROCESSING 10-20-30

Introduction

Computer processing is designed to provide all students with an opportunity to study the impact of the computer on the individual and society. Programming and software manipulation should provide students with an opportunity to direct personally the power of the computer to solve simple to complex problems.

The skills and knowledge developed will be applicable to all disciplines. Students have the opportunity to develop basic skills for the personal use of the computer; skills that support their career plans, or skills that provide suitable background for further post-secondary study.

Objectives

The student will:

1. examine the fundamental terminology, equipment procedures, and concepts used in a computerized environment

2. develop the ability to use the computer to help solve problems, manipulate data, and draw inferences from data

3. examine the potential of computers in personal and professional activities

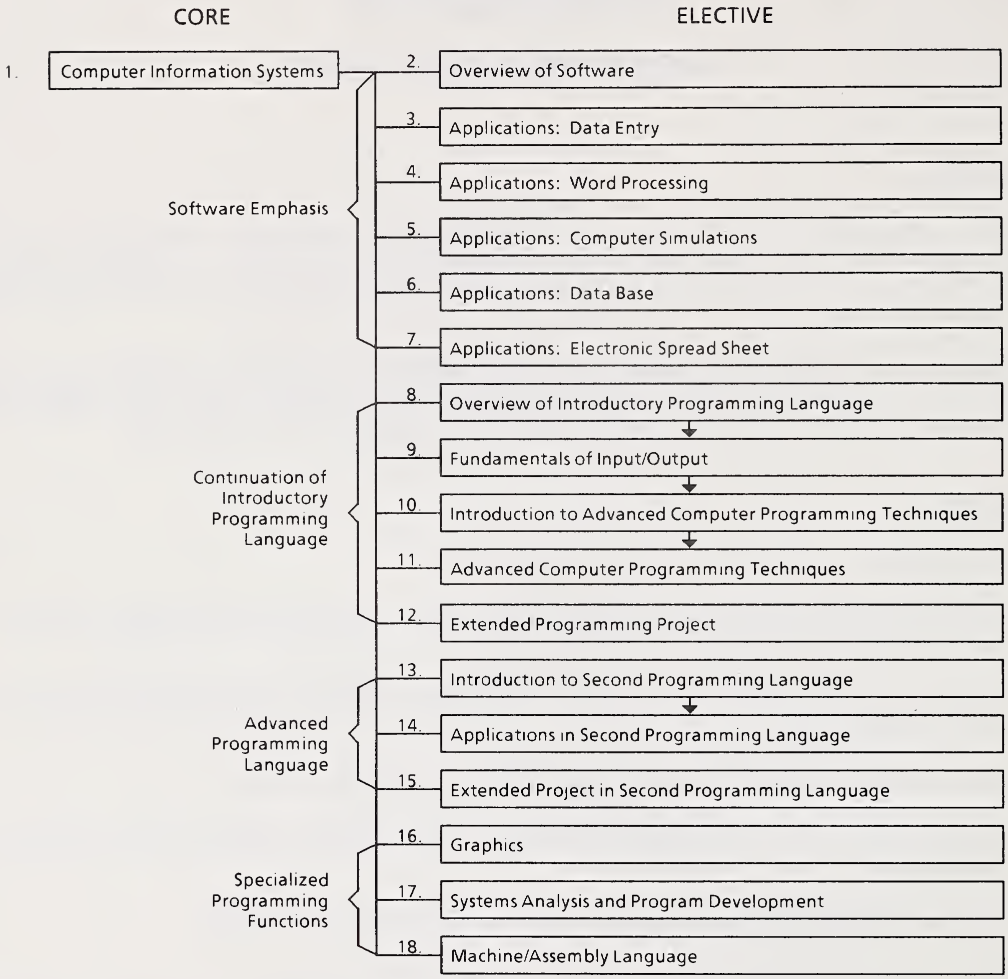
4. become adept at the manipulation of hardware and software as preparation for career or personal use

5. develop expertise in efficient programming techniques and program design

6. become aware of the potential for computer-based communications

7. recognize the impact of computers on productivity and career paths.

FLOWCHART OF MODULES: COMPUTER PROCESSING


CORE
ELECTIVE
1. Computer Information Systems
2. Overview of Software
3. Applications: Data Entry
4. Applications: Word Processing
5. Applications: Computer Simulations
6. Applications: Data Base
7. Applications: Electronic Spread Sheet
Software Emphasis
8. Overview of Introductory Programming Language
9. Fundamentals of Input/Output
10. Introduction to Advanced Computer Programming Techniques
11. Advanced Computer Programming Techniques
12. Extended Programming Project
Continuation of Introductory Programming Language
13. Introduction to Second Programming Language
14. Applications in Second Programming Language
15. Extended Project in Second Programming Language
Advanced Programming Language
16. Graphics
17. Systems Analysis and Program Development
18. Machine/Assembly Language
Specialized Programming Functions

Notes on Course Implementation

Computer processing has 18 modules, 1 core and 17 elective. Courses of 3 or 5 credits (3 or 5 modules) may be developed at the 10, 20 and 30 levels to a maximum of 15 credits. Courses will need to be scheduled in an appropriately equipped computer lab. The recommended minimum ratio is two students to one machine.

Modules 2-7 deal with software applications. Module 8 may be used to review introductory programming concepts introduced in other courses. Programming is continued in Modules 9-12, and second-language programming in Modules 13-15. Modules 16, 17, and 18 introduce specialized aspects of computer processing.

Modules from the Optional strand may be integrated into computer processing if not used previously within a student's program. It is recommended that a choice be made from the following Optional modules:

Module No.		
	10.	Business Research Project
	11.	Goal Setting and Career Development
	14.	Word Processing 1
	16.	Business Simulation

Some modules may have prerequisites and equipment requirements. Refer to individual modules for further information.

Computer Literacy 10 may be used as an alternative prerequisite to Computer Processing 20.

PLANNING

PLANNING

Long-Range Planning

The computer processing strand allows schools to design courses that are suitable for their students. Students will have a variety of interests (software applications, programming, telecommunications, etc.) and backgrounds (previous coursework in elementary or junior high school, home use, etc.).

The planning work sheets, included in the *Business Education Manual for Administrators, Counsellors, and Teachers*, may assist in the selection of modules.

The computer processing strand is composed of 18 modules, and students are eligible to take a maximum of 15 modules (or credits) throughout the three grade levels. Planning the computer processing program will involve consideration of these questions:

1. How many credits (modules) are to be offered at each grade level (three or five)?

2. How will the computer processing courses support other computer-related courses offered in the school?

3. Will the courses be designed to allow students entering the program to focus on programming? Software use? Combined programming/software?

4. What modifications will be made for students who have previous experience on computers? (Many students have worked with computers at home and developed "informal" programming or software use procedures.)

5. How are the modules that are selected to be organized within each course?

The program is also affected by:

1. the level of teaching expertise available within the school

2. the type and quantity of hardware available

3. available software and related support materials.

The flexibility of the modular format should be used to its maximum potential.

COURSE PLANNING WORKSHEETS FOR COMPUTER PROCESSING 10-20-30

Maximum Credits: 15
Course Sequence: Computer Processing 10 (3) (5), 20 (3) (5)*, 30 (3) (5)

		MODULE	COURSE LEVEL	MODULE PREREQUISITE	EQUIPMENT SUPPORT (REQUIRED/RECOMMENDED)
CORE		1. Computer Information Systems	10/20	Nil	Recommends access to suitable hardware/software support.
ELECTIVE	SOFTWARE EMPHASIS	2. Overview of Software	10/20/30	CP Module 1	Requires access to commercially prepared software packages.
		3. Applications: Data Entry	10/20/30	CP Module 1	Requires access to number pad and suitable software such as data entry, general ledger, electronic spread sheet.
		4. Applications: Word Processing	10/20/30	CP Module 1 TY Module 3	Requires suitable word processing software. Recommends additional printer support.
		5. Applications: Computer Simulations	10/20/30	CP Module 1	Requires suitable software and hardware support.
		6. Applications: Data Base	10/20/30		Requires access to data base and/or related software. Recommends access to modem.
		7. Applications: Electronic Spread Sheet	10/20/30		Requires access to electronic spread sheet software.
	INTRODUCTORY PROGRAMMING	8. Overview of Introductory Programming Language	10/20	CP Module 1	Requires suitable hardware support.
		9. Fundamentals of Input/Output	10/20/30		
		10. Introduction to Advanced Computer Programming Techniques	10/20/30	CP Module 9	
		11. Advanced Computer Programming Techniques	10/20/30	CP Module 10	
		12. Extended Programming Project	10/20/30	CP Module 1	

		MODULE	COURSE LEVEL	MODULE PREREQUISITE	EQUIPMENT SUPPORT (REQUIRED/RECOMMENDED)
ELECTIVE	ADVANCED PROGRAMMING	13. Introduction to Second Programming Language	20/30	CP Module 1	Requires suitable hardware support and facility to handle the designated programming language
		14. Applications in Second Programming Language	20/30	CP Module 13	
		15. Extended Project in Second Programming Language	20/30	CP Module 14	
	SPECIALIZED PROGRAMMING	16. Graphics	10/20/30	CP Module 1	Requires suitable hardware support. Recommends access to graphics tablet, suitable printer support.
		17. Systems Analysis and Program Development	20/30	CP Module 1	
		18. Machine/Assembly Language	10/20/30	CP Module 1	Requires suitable hardware support.

A maximum of one optional module may be included in a three-credit course,
or two modules in a five-credit course.

OPTIONAL	10. Business Research Project	10/20/30		
	11. Goal Setting and Career Development	10/20/30		
	14. Word Processing 1	10/20/30		
	16. Business Simulation	10/20/30		

* Computer Literacy 10 may also serve as a prerequisite.

Short-Range Planning

When specific modules have been identified for each grade level, decide the sequence in which the modules and module topics will be presented.

Not all modules have to be presented in the same sequence as they appear in the curriculum guide, and topics within each module may be rearranged within the module or interspersed throughout the course.

Step 1 Study the contents of the appropriate module(s) in the *Computer Processing 10-20-30 Curriculum Guide* and the "Teaching Suggestions" section of this manual. Compare topics to be covered with the identified texts or other sources of information that are available. Also consider students' needs and what facilities are available.

Step 2 Decide upon a sequence for the topics.

Step 3 Determine specific lesson content for each topic in the module(s). Identify specific objectives, teaching strategies, and appropriate evaluative procedures. Enunciate expectations in terms of competency levels and decide what learning domains are to be involved.

Step 4 Ensure that all required resources (audio-visual resources and equipment, supplies, software and documentation) are available when needed, and that field trip arrangements are confirmed.

Step 5 Prepare appropriate evaluation instrument(s) and/or identify evaluative processes that will accurately measure attainment of the objectives.

Note: Hardware and Software Considerations

Computer processing cannot be taught without access to functioning hardware and documented software. Supplies of printer paper and disk backup must also be available at all times. Appropriate security measures and adequate software storage should be incorporated. Student demand or need to use computer equipment may require that these facilities be available throughout the entire day. Since students will often work on their own, teachers must be familiar with the software packages being used and anticipate potential problem areas.

SAMPLE LESSON PLANS

SAMPLE LESSON PLAN 1

MODULE 1
COMPUTER INFORMATION SYSTEMS

TOPIC: Characteristics of Information Systems
(Pages 1.1 - 1.5 -- Shelly & Cashman)

TIME: 50 minutes

Objective At the conclusion of this lesson students will be able to:

1. describe what a computer is and how it works
2. accurately label each component part of the three primary units of a computer system
3. define and explain the computer and data processing terms that were used.

Background Information

This lesson introduces students to the basic elements of a computer, how the computer processes data, and how computers can function as data processing systems.

Advanced Planning Materials

Teacher plans could include:

- identification of specific content, sequence or presentation, and information sources
- methods of presentation - motivation, explanations, visual aids, type and amount of student participation and student activities
- physical arrangement of hardware and students during class demonstration of computer operation and the computer architecture
- evaluation of student comprehension.

CATEGORIES	STRATEGIES	STUDENT ACTIVITIES
Introduction (10 minutes)	Briefly demonstrate to the class the operation of an interactive program (teacher-developed application or commercially prepared software package). Ask questions/explain what is happening.	observe ask questions answer questions
Development (25-30 minutes)	Develop the concept of a computer through discussion. (Text examples – pp. 1.0, 1.2, 1.3.) As new terms are encountered, write them on the blackboard and discuss.	answer questions refer to text take notes
	How do computers work? Solicit student responses. (Text example – 1.5, Transparency-T1.) Describe and discuss basic computer system(s).	explain observe copy and label diagram
	Demonstrate and discuss details involving another interactive program – perhaps GPA program on p. 1.6 of text. Explain: input raw data (source document) output information (hard/soft copy) Compare program output with Transparency-T2.	observe comment take notes ask questions
	Discuss role of software (computer program). Explain input/processing/output sequence. Text illustration and explanations pp. 1.7-1.15, Transparency-T3, magazine articles, pictures, diagrams, sample programs, source documents, hard copy output, etc.	take notes ask questions comment observe

CATEGORIES	STRATEGIES	STUDENT ACTIVITIES
	Show class the inside of a computer and explain its various parts.	observe discuss
Closure (5 minutes)	Orally summarize computer architecture and how it works or demonstrate another program and have students describe what is happening.	answer questions describe
	(Ask students to supply materials – examples, pictures, diagrams – of computers, source documents, and output samples for bulletin board display.)	take notes
Feedback (5-10 minutes)	Oral or written review of: 1. what a computer is 2. how a computer works 3. definition of terms 4. labels on computer diagram—either on blackboard, transparency, handout sheet or workbook (p. 1.5).	respond orally in writing

Evaluation

Student

- quality of responses and comments
- willingness to respond or volunteer questions or comments
- type of questions or comments offered
- amount of student interest and interaction

Teacher

- did demonstrations help obtain student interest and provide a clearer presentation?
- what activities/topics seemed to be understood/unclear?
- were time allotments appropriate?
- were resource materials effective?

Follow Up

This lesson is designed to provide basic information about computer architecture and introduce the various types of computer systems – data processing systems, management information systems, file processing systems, etc.

The lesson might be followed up by creating a bulletin board display of computer information and materials, using the contributions of the teacher and the students.

Students could also participate in oral reports, written reports, student debates, or class discussion (to help develop communication skills on a variety of computer-related topics as well as develop better familiarity with computers), student demonstration of computer applications, or a class tour of a computer installation.

SAMPLE LESSON PLAN 2

MODULE 4
APPLICATIONS: WORD PROCESSING

TOPIC: Introduction to Information Processing

TIME: 50 minutes

Objective At the conclusion of this lesson students will be able to:

1. describe the purpose of word processing hardware and software
2. boot the word processing software
   enter data or text, either by using the keyboard or loading from an existing file.

Background Information

This lesson is the introductory module for word processing. It introduces students to the advantages and limitations of using specialized hardware and software to enter and manipulate text.

Teachers can use three approaches to this introductory lesson:

1. Conceptual discussion about word processing, word processing software and hardware, and word processing applications.
2. Practical demonstration of how one specific word processing package works ("hands-on" application at a very elementary level).
3. Combination of some conceptual discussion combined with demonstration and some very limited "hands-on" experience for students.

Advanced Planning Materials

Teacher plans could include:

- identification of a specific approach
  (conceptual versus practical) and specific objectives

- method of presentation - motivation, visual aid support, samples of output from word processors, student involvement and activities

- physical arrangement of hardware and students during computer demonstration of word processing functions

- if student "hands-on" experience is selected, plans as to who boots the system, how many software disks are needed, and type of input anticipated

- determination of student feedback.

CATEGORIES	STRATEGIES	STUDENT ACTIVITIES
Introduction (5 minutes)	Show various types of output using left, right, and center justification. Briefly discuss typewriter versus word processing capability. Ask for students' word processing experience.	observe ask questions answer questions discussion
Development (40 minutes)	Demonstrate (with appropriate explanation) the following commands: load, cursor control, text entry via keyboard and from file, some data manipulation using "insert" and "delete".	observe ask questions answer questions
Closure (5 minutes)	Students boot word processing software and try cursor control, keyboard text entry, file entry from disk, and limited text manipulation.	"hands-on" practice ask questions comments react
Feedback	Give oral summary of commands and how each command is executed. Direct questions to class.	respond to questions ask questions
	Continually monitor students throughout the entire class period. Consider questions and answers received from students, ability of students to proceed with various commands, student demonstration of degree of confidence; e.g., body language.	

Evaluation

Student

- quality and type of questions and comments
- willingness to communicate orally and frequently
- willingness to experiment with commands
- number of commands covered

Teacher

- did the introduction provide appropriate setting?
- were demonstration and explanations clearly understood?
- did "hands-on" activities accomplish objectives?
- were time allocations appropriate for each activity?
- were students able to execute identified word processing commands?

Follow Up

This introductory lesson is designed to familiarize students with word processing as a tool to produce hard copy output in an acceptable format.

Most students will exhibit considerable interest in the software and its potential. One possible area of concern is that students who do not possess reasonably efficient keyboarding skills definitely will be at a disadvantage when entering text. There are individualized computer-based keyboard training programs, but students need to have access to the equipment and must be prepared to spend some extra time in developing skill on their own.

Obviously the more work stations in the classroom, the more difficulty the teacher experiences in attempting to provide individual attention and adequately monitor individual student's progress.

The challenge for the teacher is to encourage all students to realize their potential to develop their skills and to inspire them to use the word processing equipment for personal needs whenever they can, regardless of what their original keyboarding skill levels may be.

SAMPLE LESSON PLAN 3

MODULE 8
OVERVIEW OF INTRODUCTORY PROGRAMMING LANGUAGE

TOPICS: a. Introduction to Entire Module
b. Programming Techniques

PURPOSE OF LESSON: To introduce programming, and programming languages.

TIME: 60 minutes

Concept: (5 min.) Purpose of All Languages

Activity: Ask students to explain the purpose of language. Students may have to be directed to think about language in general – not just computer language, but "body language" could also be introduced (e.g., OK signs, thumbs up, etc.).

Concept: (15 min.) The Nature of Language – Syntax and Semantics

Activities:

1. Use a sentence that is syntactically incorrect but interpretable – "when me using speaking what ain't much goodly yous still knowed what I has been meaning." – then ask students to explain its meaning, and if they can tell you what is wrong with the sentence.

2. Explain that the grammar of a language is defined by its "syntax", which describes the way in which valid statements can be constructed (e.g., adjective before noun in "the yellow bus", and not "the bus yellow").

3. Using the overhead projector, examine a "nonsense" sentence that COULD be syntactically correct. (For example, "Twas brillig and the slithy toves did gyre and gimble in the wabe" is syntactically correct if "brillig" and "slithy" are adjectives, if "toves" and "wabe" are nouns, and if "gyre" and "gimble" are verbs.) Explain that the meanings of words and, therefore, sentences are determined by "semantics" and that this sentence is semantically incorrect because these words have no English meanings. Point out that computer languages also have the characteristics of syntax and semantics.

Concept: (10 min.) Language Processors as a Bridge Between Machine and Man

Notes:

The "native" language of machines is called machine language, an electronic language of voltages and no-voltages which programmers and electronics people represent by 1's and 0's. Each type of central processor has its own machine language, just as each country has its own native language. Humans do not communicate well in computer language (1's and 0's), just as English-speaking people do not communicate in Swahili.

The first step in the development of machine language was the use of a mnemonic, a word or sequence of letters that suggested and stood for the action performed by the 1's and 0's. For example, ADD was used to represent the 1's and 0's that caused an add to occur in the computer. This was the birth of Assembler language programming. The programmer types "ADD" and the computer translates it to 1100 0110 (or whatever the computer uses for addition). This is referred to as a "low-level language": one Assembler instruction yields one machine language instruction.

The next step was to use a translator program that could convert one mnemonic into the sequence of several machine language instructions required to achieve it. For example, the mnemonic PRINT was converted into the many individual machine language instructions that are required to get some information on the screen. These translations are referred to as "high-level languages". FORTRAN, developed in 1957, was the first successful high-level language. BASIC is also a high-level language.

Language processors (programs that convert English-like words into computer language), act like translators at the United Nations, taking the representation of information in one form and converting it for use in another form.

Concept: (20 min.) Comparison of Computer and Natural Language

Activities:

In this part of the lesson, only students who have some background will likely be able to answer the questions. Point this out, so that inexperienced students will feel comfortable about taking part in the discussion.

Criterion 1 - Number of "words" in a language.

Ask about the number of words in computer language as compared to natural language. (Computer languages have only a limited number of words whereas natural language has many.)

Show overhead transparency of the instruction set of the computer language to be used. Compare that with the number of words in a dictionary.

Criterion 2 - Tolerance for ambiguity.

Ask about the degree of ambiguity allowed in natural language as compared to computer language. (A great deal in natural language, very little in computer language.)

If answers or discussion are not forthcoming, ask for the answer to a statement such as "Where is West Edmonton Mall?" If you are in Calgary, the answer might be "In Edmonton"; if you are in Ottawa the answer might be "In Alberta"; if you are in Edmonton, the answer might be "On 170 Street".

Criterion 3 - Variation from syntactic rules allowed.

Ask about variation allowed in computer language as compared to natural language. (Natural language has a great deal of variation.)

If necessary, remind students of the grammatically incorrect statement used at the beginning of class.

Criterion 4 - Richness of expression.

Ask students how well computer languages can be used to express feelings, emotions, and other complex human characteristics.

Concept:
(5 min.)

Computer languages serve certain types of uses, while natural language serves others.

Activities:

Ask which is the better, computer languages or natural language?

Concept:
(5 min.)

Some computer languages are better than others for certain uses.

Activities:

Explain that some natural languages are better than others for expressing certain concepts. For example, the Inuit language has many words to describe different types and textures of snow because it is important for Eskimos to be able to describe snow accurately. Likewise, some computer languages are graphics oriented, others are file oriented, and so on.

Conclusion of Lesson:

Conclude by telling students that the BASIC language is relatively easy to learn and is widely used on microcomputers.

TEACHING SUGGESTIONS

TEACHING SUGGESTIONS

MODULE 1: COMPUTER INFORMATION SYSTEMS

Status: Core

Purpose

The purpose of this module is to introduce the student to the world of computerized information processing. This is the only core module in the computer processing strand. It is designed to provide an overview of the computer – what it is, how it works, and how it can be used to aid in a variety of problem-solving situations.

This introductory module includes a brief summary of the historical aspects of computer development, a concise description of computer system architecture, and related computer industry terminology. Topics include the evaluation of data processing from a development/historical perspective; characteristics of computerized information processing systems (including the data processing cycle of input, processing, and output; data flow within computer systems; the relationship of hardware, software, firmware, and computer peripherals); information processing environments, current issues and trends faced by society, and data communications – within the department, with the computer itself, and with computerized systems and networks.

The need to incorporate professional standards and attitudes when handling computer hardware, software, data, and information raises several controversial issues such as responsibility, privacy, integrity, and reliability. Also, continuing technological developments in the computer industry will produce more sophisticated hardware and more "user-friendly" integrated software. The future results of this advancing computer technology will be to provide expanded opportunity for personal use in the home (Telidon), business use in the office (electronic mail), educational use in the school (CAI), plus a wide range of computerized applications in industry (CAD-CAM, simulations, and robotics).

Planning

There is no one "best approach" to presenting information in a classroom. However, it is suggested that the various topics in this module be integrated throughout the course by interspersing them with "hands-on" activities from other modules. This approach enables students to experience a variety of activities without unduly prolonged discussion at any one time of the theoretical aspects of the material being covered. Whenever possible, actual classroom computer demonstration should be incorporated to help illustrate the theoretical topics under discussion.

This module also provides an opportunity for students to become involved in research projects, oral or written reports, discussions, or bulletin board displays. The challenge is to develop a learning environment that will encourage meaningful student involvement and participation in the exploratory stages of this computer course.

Resources

Some overlap of topic content and page references may occur. Some of the content may also be found in resources other than those identified.

Each chapter in the Shelly & Cashman text includes a Chapter Summary, Student Learning Exercises, Controversial Issues in Data Processing, and Suggestions for Research Projects. In addition, the publisher provides the following support material:

a) Instructor's Guide and Answer Manual – contains numerous suggestions regarding the presentation of content and follow-up activities.

b) Transparency Masters – correlated with the text.

c) Test Bank – correlated with the text, includes numerous true/false and multiple choice questions for each chapter, including the appendices portion of the textbook.

d) Student Workbook – correlated with text, provides several project suggestions for each chapter – application projects and exercises as well as review and quiz exercises.

Other readily available sources are hardware/software vendors and locally developed materials.

Evaluation

This module basically comprises a conceptual approach to the study of computers. As a result, evaluation will essentially be composed of theory questions. There should be a reasonable balance between the lower end of the evaluative continuum (involving questions of recall of information only), and the higher level evaluative continuum (involving questions designed to require analysis, synthesis and application). Suggestions as to time allocation, amount of detail, and evaluation emphasis are included in the topic summary pages. Evaluation methods should directly reflect the objectives and the degree of emphasis placed on the topic.

For more specific details about evaluation, refer to Part IV: "Student Evaluation" of the *Business Education Manual for Administrators, Counsellors, and Teachers.*

Theoretical	100%

MODULE 1 -- COMPUTER INFORMATION SYSTEMS

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
EVOLUTION OF DATA PROCESSING Shelly – Chapter 2 (2.0-2.31) Sanders – Chapter 1 (31-35) – Chapter 15 (391-422) – Chapter 16 (423-451) – Chapter 17 (452-473)	15% (T)	Have students work in groups developing timelines/charts. Present films/filmstrips depicting historical development. Assign reports (oral or written) to develop various historical aspects of interest.	(Theory - 100%) 1. Reports – oral written 2. Quiz a) Objective – multiple choice – fill-in blank – label diagram – matching b) Subjective – definition – short answer – paragraph
TERMINOLOGY Mandell – Chapter 1 (21-22) Sanders – Chapter 1 (29)		Introduce and discuss terms as they are encountered, and present an oral or written definition review. Basic terms should include at least: computer, cpu, data, hardware, input, information systems, memory, output, processing, peripheral devices, software, firmware, RAM, ROM, disk.	
CHARACTERISTICS OF INFORMATION SYSTEMS Shelly – Chapter 1 (1.0-1.33) – Chapter 2 (2.01-2.49) Sanders – Chapter 1 (8-14) – Chapter 2 (36-62) – Chapter 12 (305-332) – Chapter 14 (357-390)	35% (T)	Use the following to enhance presentation: – transparencies – films/filmstrips on computer systems/architecture – pictures, magazine articles – samples of source documents and hard copy output – advertising brochures Arrange a field trip to a computer installation. Assign reports (oral or written) to cover various aspects of information systems.	
INFORMATION PROCESSING ENVIRONMENT Shelly – Chapter 1 (1.7-1.29)	15% (T)	Demonstrate application software capability, and have students describe various environments. (See field trip suggestions.) Use the following to enhance presentation: – pictures from trade magazines – bulletin board displays	

NOTE:

(T) Concept lends itself to theoretical presentation.

(P) Concept lends itself to practical applications.

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
INFORMATION PROCESSING ENVIRONMENT (cont'd) Sanders – Chapter 1 (14-23) – Chapter 5 (97-125) – Chapter 7 (170-205) – Chapter 8 (206-230) – Chapter 9 (231-250) – Chapter 13 (334-344)		Enhance topic presentation with film/filmstrips, magazines, transparencies, and/or pictures. Arrange a field trip to a computer centre, a computer vendor, or a large data centre of a business. Conduct an in-class discussion, assign student research projects/reports (oral or written), and/or use teacher-directed activities to develop the topic.	
DATA COMMUNICATIONS Shelly – Chapter 2 (2.25) – Chapter 8 (8.0-8.31) This topic is also covered in Module 9. Sanders – Chapter 10 (251-285) – Chapter 11 (295-304)	15% (T)	Use transparencies, pictures, and magazine articles to supplement concepts being presented. Arrange field trip to: – travel reservation office – CN/CP railway switching centre – department store – cable company – post office Invite a guest speaker from the industry to discuss data communication. Conduct an in-class discussion on the topic, or assign reports (oral or written) to develop specific areas of interest.	
ISSUES AND TRENDS Shelly – Chapter 2 (2.26-2.28) – Chapter 13 (13.0-13.8) Sanders – Chapter 4 (78-92) – Chapter 15 (391-422) – Chapter 17 (452-474)	20% (T)	Present films on robotics and future trends. Assign reports (oral or written) to develop specific areas of interest. Invite a guest speaker, and/or conduct an in-class discussion/debate. Supplement learning resource material with magazine and newspaper articles.	

MODULE 2: OVERVIEW OF SOFTWARE

Status: Elective
Prerequisite: Module 1

Purpose

In this module, students have an opportunity to become familiar with the characteristics and appropriate use of various types of software (including the proper handling, storage, and hardware application constraints). They also have an opportunity to use a variety of software packages in problem-solving situations.[1]

Planning

Students must learn the proper handling techniques and the appropriate use of software. They need to know not only about the physical handling and storage of the software packages, but also the legal constraints of copyright[2], the need to match hardware and software specifications appropriately, and the proper application and limitations of each software package.

The students should be introduced initially to software that is reasonably "user-friendly" and supported by easily understood user documentation or instructions. Students are faced with two problems when they first encounter a software package—mastery of the commands of that package, and the ability to use the commands in a meaningful problem-solving application. Therefore, the coded instructions required to operate each package should be simplified whenever possible. For example, summary instructions can be placed on the blackboard, transparencies, and handout sheets, or provided through oral discussion and dictated notes in addition to the publisher's documentation.

Many times, a teacher can prepare simple application problems beforehand and store them on a disk. Students then load this application data into their own computers and immediately begin manipulating the data using the software package. This approach can save considerable student keyboard entry time and frustration, particularly if students lack efficient keyboarding skills. Once students have developed sufficient operation or manipulation skill, they can be encouraged to proceed with their own data entry via the computer keyboard.

---

1. Business Education Manual for Administrators, Counsellors, and Teachers discusses the appropriate software integration in the various strands and modules of the Business Education program.

2. Ibid., page 81.

Resources

The following are suggested software applications (which may be effectively incorporated in such areas as business, chemistry, language arts, mathematics, and physics):

a) utility programs - normally included with the specific computer system being used, perhaps included on the master disk (diagnostic, renumbering, formatting, copy and backup, etc.)

b) application software - can be used for problem solving or practical use (word processing, electronic spread sheet, data base, etc.)

c) simulation software - wide variety of applications in problem-solving area, including educational "games"

d) drill and practise - wide range of applications providing various levels of difficulty in problem-solving area

e) computer assisted instruction - designed to provide an interactive individualized approach to student learning, reinforcement, or review

f) authoring systems - a simplified method of generating programming instructions through the use of menu-based word processing oriented software. The user is provided with step-by-step prompts in creating input, formatting output, and while interacting with the computer system.

Evaluation

The majority of this module is devoted to "hands-on" skill building and application activity. Therefore, the evaluative process (theory and practical) should reflect this. Students can be provided with a problem and asked to solve it completely, or they may be provided with some of the data on disk and proceed directly to the manipulation/processing stage. It is also possible to use a theory test (multiple choice or fill-in) to test knowledge of software/hardware commands as well as observing the actual manipulative process and/or output.

Students should have both formative and summative evaluations as they progress through this module. The actual form of evaluation will normally depend upon the available physical facilities, number of students involved, goals, and philosophy of the teacher.

Theoretical 25%	Practical 75%

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
TERMINOLOGY		Introduce and discuss each term as encountered. Present oral or written definition reviews. Basic terms should include at least: hardware, application software, interactive software, system software, copyright, firmware, disk operating system(DOS), stored program concept, programming languages, and computer systems.	1. Theory tests (25%) a) Objective – multiple choice – fill-in blank – matching b) Subjective – definitions – short answer – paragraph 2. Practical "hands-on" tests (75%) a) Demonstration of software/hardware manipulation b) Problem-solving applications using software packages
LEGAL CONSTRAINTS TO USE OF SOFTWARE – revised statutes of Canada Mandell – Chapter 1 (14-15)	5% (T)	Discuss: – copyright laws of Canada – publisher disclaimers – public domain software – legalities of copying software	
PROPER HANDLING TECHNIQUES – operator manual Mandell – Appendix H (408)	10% (T/P)	Explain and demonstrate proper techniques for handling and storing software, using pictures, transparencies, or bulletin board displays to enhance presentation. Observe student technique. Discuss and demonstrate methods of evaluating software programs -- limitations, potential, hardware specifications.	
TYPES OF SOFTWARE Shelly – Chapter 1 (1.24) – Chapter 2 (2.10, 2.12, 2.19, 2.21) Sanders – Chapter 8 (221-226) – Chapter 11 (286-294) – Chapter 12 (295-304) – Chapter 13 (333-356) Mandell – Chapter 1 (12-14)	10% (T/P)	Discuss, explain, and demonstrate the use of various software packages. Observe student technique in using software packages. Suggested software: – word processing – electronic spread sheet – accounting – data base – integrated – CAI – locally developed (Try to employ user-friendly, interactive software.)	

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
SOFTWARE APPLICATIONS Shelly – Pages 1.24, 2.19, 12.16* Individual Software Package Documentation Sanders – Chapter 8 (216-230)	10% (T/P)	Suggested activities could include one or more of the following packages: – word processing: text creation, editing – spread sheet: table revision/arithmetic calculations – accounting: posting to accounts, financial statements, budgeting – data base: data storage, manipulation and retrieval – integrated software: composing, editing, merging, outputting information (text and data) – CAI: drill and practise – locally developed: could include personally created programs or modifications to existing programs, including commercial software	
SOFTWARE-USE PROCEDURE (Several application texts are available for most packages, or provide teacher-designed applications.)	65% (T/P)	Explain and demonstrate potential applications of each of the various software packages. Provide sufficient "hands-on" opportunity, to obtain minimum skill levels on basic commands of several software packages and to use these available packages to solve assigned problems.	

*Chapter 12 of the text deals with programming languages.

MODULE 3: APPLICATIONS: DATA ENTRY

Status: Elective
Prerequisite: Module 1

Purpose

In this module, students gain a conceptual base for preparing and coding raw data for entry into computer systems, and they develop keyboarding skill in the data entry process. More specifically, in this module students study the various types of devices and methods used to enter data into computer systems; develop skill in efficiently preparing and coding raw data for the input process; further develop keyboard skills on the alphabetic, numeric, and function keys of the computer; develop touch stroking proficiency on the 10-key numeric pad; and refine stroking skill in entering data (alphabetic, numeric, and alphanumeric) into a computer.

Planning

The computer system must be equipped with a 10-key number pad in addition to other hardware configurations already identified for the computer processing strand. If there are not sufficient computers, students may develop initial touch stroking proficiency on a 10-key number pad by practising on electronic calculators.

The theory topics in this module should be interwoven with the "hands-on" skill building practise portions. While the students progress through the various stages of skill development on the keyboard, they also cover the theoretical aspects of data entry. After initial "hands-on" skill development has occurred, and by the time the students are ready to begin the actual coding of the raw data for entry into the computer system, they will have obtained the necessary background to complete their application projects efficiently. A variety of learning experiences is provided in this approach.

In the classroom, students will develop "hands-on" skill in working with specific hardware and specific machine-dependent software that may or may not be duplicated in the business environment. Therefore, if students are going to adapt successfully to different hardware and software configurations once they leave the school setting, they must thoroughly understand the principles behind the preparation of source documents for the data entry process.

To help students to understand the data entry process, generic terminology as well as software-specific terminology should be incorporated in this module. Transparencies can help illustrate to the students such things as diskette mapping and data organization––the relationship of bit, byte, character, field, record, file, sector, and data base. Transparencies can also be used to explain graphically, through a flowchart format, the data cycle—the flow of data from its creation on a source document to its final disposition in a department file.

A variety of approaches can be used to introduce this module. For example: a conceptually based class discussion of how source documents are prepared and how data are coded for entry into a computerized system; teacher demonstration of actual data entry techniques; data manipulation of a stored data file using the software package; hard copy listing of stored data or a printout of processed information; actually "walking through" a typical example with the class while they duplicate each step on their own systems; or any combination of these.

Resources

In addition to the 10-key number pad, students will need access to a data entry software package and practise drills and exercises. Ideally students will have access to software that is user-friendly, and programmed to identify the data entry format, verify the accuracy of the keystrokes (numeric, alphabetic, or alphanumeric), and tally the strokes and errors made by the data entry operator. Less elaborate software packages may include only some of the above characteristics. Also, some software packages include hard copy correlated drills, exercises, and simulated projects for student practice and problem-solving applications.

If a data entry text is being used for this module, practise drills and a variety of exercises for coding and entry will normally be included. Teachers may wish to develop their own drill and/or practise material either to replace or supplement an existing data entry text. For example, teachers could use numeric drill exercises from existing 10-key calculator texts to help build skill on the number keypad.

Evaluation

Evaluation should include the measurement of the conceptual principles as well as "hands-on" practical application of data entry. Theory and application questions can be used to measure student achievement of the conceptual principles, while "hands-on" timings and/or projects can be used to measure the level of psychomotor skill development. Keyboarding skill competency levels should include both speed and accuracy. Obviously, expected performance levels in business will vary depending upon the specific application. Keystroking speeds will to some extent be influenced by the specific hardware configuration being used.

The usual guideline for measuring data entry performance on the keyboard is identified in keystrokes per hour. Typical acceptable industry expectations for data entry involving keyboard-to-diskette configurations ranges from 8,000 to 14,000 keystrokes per hour with a maximum error rate of four percent. For the typical classroom, rates could range from 2,000 to 8,000 keystrokes per hour on ten-minute timings of straight copy input data (numeric or alphanumeric), with a five percent error tolerance before penalty. Some software packages automatically count and verify the accuracy of the strokes as they are entered.

If appropriate testing material is not included in the data entry text or is not readily available, locally developed evaluative instruments will adequately measure the achievement of the objectives.

Theoretical 25%	Practical 75%

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
DATA ENTRY FUNCTIONS Shelly – Chapter 3 (3.12-3.27) – Chapter 5 (5.1-5.37) – Pages 10.11, 10.19, 10.22 Sanders – Chapter 6 (129-169) – Chapter 15 (421-422)	30% (T/P)	Discuss data entry department and its functions. Demonstrate and explain proper touch-typewriting techniques using the standard microcomputer keyboard and number pad; supervise student practise on both keyboards. (Use CAI keyboarding software, if available.) Provide directed drills and timings (alphabetic, numeric, alphanumeric) from typing texts, typing drill books, or data entry texts for maximum student skill development. Demonstrate and explain: – data entry software package – word processing software package – accounting software package Supervise student practise on data entry packages, and possibly other available packages.	1. Theory tests (25%) a) Objective – multiple choice – fill-in blank – matching b) Subjective – definitions – short answer – paragraph c) Report – oral or written
CODING DATA Shelly – Chapter 5 (5.1-5.9, 5.20-5.22) Sanders – Chapter 2 (37-38) – Chapter 6 (129-169)	10% (T/P)	Discuss: – source documents (examples) – the need to prepare data from source documents for data entry – types of documents (internal, external) e.g., orders, invoices, inventory vouchers, payroll, accounts receivable, accounts payable Demonstrate, explain, and have students practise the proper coding of data on various types of source documents. (Source documents can be provided by the teacher, students, or from data entry texts.)	2. Practical "hands-on" tests (75%) a) Demonstration of software/hardware manipulation b) Problem-solving application from source documents using software packages
DATA ENTRY Shelly – Pages 1.16, 2.24, – Chapter 5 (5.1 - 5.37) Sanders – Chapter 6 (129-169)	20% (T/P)	Discuss types of data entry, specific hardware, and physical location. Have students practise data entry skills using alphabetic, numeric, and alphanumeric input data.	(Evaluation should include speed and accuracy.) General Guidelines for Key/Disk Operation 2,000 - 8,000 keystrokes/hr. on a 10-minute timing from straight copy numeric or alphanumeric data with 5% error tolerance. (Letter grades or percentages can be assigned for both speed and accuracy.)

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
GENERATING REPORTS Shelly – Chapter 6 (6.6-6.15) Sanders – Chapter 7 (187-188)	40% (T/P)	Discuss report considerations - type (hard or soft copy), style, format, physical characteristics, and intended use. Have students: – code raw data on source document(s) – enter data using software packages – use software commands to manipulate data – output information in acceptable format [Several exercises or projects on a variety of activities can be developed by the teacher or obtained from data entry text(s).]	

MODULE 4: APPLICATIONS: WORD PROCESSING

Status: Elective
Prerequisite: Module 1

Purpose

In this module, students are introduced to the concepts involved in word processing, and they develop introductory keyboarding skill in the use of a word processing software package. More specifically, students study the function(s) of word processing, develop basic skill in several operation commands, and apply this knowledge to problem-solving applications.

Planning

Students must not only develop functional operation skill(s) on a specific software package and hardware configuration which happens to be in a particular classroom; they must also understand the concepts underlying the use of word processing as an aid to text creation. Therefore, students should have the opportunity to discuss, in generic terms as well as specific software dependent commands, the various functions that word processors are capable of performing. If this approach is undertaken, students should be able to transfer their school-developed skill to other word processing systems outside of the school environment.

Regardless of the specific word processing package used, students should develop functional skill in at least the following areas: cursor control; text manipulation (creation, insertion, deletion, and replacement); file handling (retrieval, storage, merging, and deletion); and document output (text formatting and printer commands).

Ideally, all students enroled in this module will have developed a minimum skill level of "touch" keyboarding proficiency before they attempt to use a word processing package. In fact, some students may possess considerable keyboarding skill. Students should continue to improve their proficiency on the keyboard while also developing functional word processing skill.

Individual preference and classroom circumstances will probably dictate the approach used to introduce students to a word processing package. For example, the teacher could introduce this module by using a conceptual approach (class discussion about the role of word processing in textual communication); a "hands-on" demonstration involving several word processing applications (text creation and/or document output); text manipulation (editing an existing file); or, by providing the student with actual "hands-on" experience (teacher demonstrates and explains while actually "walking through" each step with students, as they replicate the steps on their individual systems).

Resources

Each type of word processor has individual characteristics. There are dedicated stand-alone word processors and shared-logic dedicated systems, in addition to a variety of microcomputers that are capable of supporting several word processing packages. Typically, word processing instructions are menu driven and/or command driven depending upon the specific characteristics of the software being used. In addition, some word processing packages contain only a limited number of basic commands and tend to be relatively easy to master, while other packages tend to be less "user-friendly", contain many commands, and take longer to master. Obviously, the more powerful and sophisticated word processing packages contain a greater range of available options.

Each word processing software package contains its own documentation; however, teachers may also wish to use either a generic word processing text or a software specific text for supplementary reference. In addition, teachers could incorporate practise exercises from texts for other courses (Office Procedures, Typing, or Word Processing 30) or use locally developed materials for their classes.

Evaluation

Theory tests can be used to test for knowledge of word processing concepts and commands. "Hands-on" practical tests can measure operational and application mastery of the word processing system. Students can be required to compose their own text input using the word processing system. They can be given straight copy or rough draft copy to be entered into the processor, or they can be asked to load a text file from disk and either proofread the copy and make any desired changes or follow instructions in order to produce the desired output. The evaluation process should reflect such aspects as speed, accuracy, format of the completed document.

Theoretical 25%	Practical 75%

COURSE: COMPUTER PROCESSING
MODULE 4 -- APPLICATIONS: WORD PROCESSING

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
INTRODUCTION TO INFORMATION PROCESSING Shelly – Pages 3.14-3.15 Sanders – Chapter 2 (56-58) – Chapter 11 (289-294) – Chapter 14 (375-376)	15% (T)	Introduce and discuss each term (generic and software specific) as encountered throughout the module including the hardware and software industry's "buzz" words. Discuss word processing in terms of: – the role of word processing in written communications (business and personal use) – efficient information handling (text creation, manipulation, output, storage, and retrieval) using electronic equipment within the office setting – hardware configurations (stand-alone, shared-logic, and multi-purpose word processors) – career opportunities (operator level to administrative/management positions) Supplement presentation with film, pictures or advertising information on the hardware configurations, output examples, actual classroom demonstrations, transparencies, word processing sales representative as a guest, or field trip to typical office or word processing vendor.	1. Theory tests (25%) a) Objective – multiple choice – fill-in blank – matching b) Subjective – definitions – short answer – paragraph 2. Practical "hands-on" tests (75%) a) Demonstration of software/hardware manipulation
SOFTWARE COMMANDS Software Documentation Word Processing Text	30% (T/P)	Explain, illustrate and demonstrate the use of word processing commands (menu or command driven): – operation commands – load from disk – save to disk – create text – manipulate text (insert, delete, move) – print text Observe and supervise student operation. Identify hardware/software configuration requirements. Basic terminology should include at least: word processing system, word processing commands (text creation, cursor control, scrolling, insertion, deletion, formatting, move, edit, merge, pagination, output, load file, save file, dictionary), information processing.	b) Problem-solving applications using word processing software (Students can create text from scratch, proofread rough draft copy and enter text, or load text from file and proofread, and/or follow instructions.) Grading should recognize speed, accuracy, and style of output documents.

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
INPUT/OUTPUT	20% (T/P)	Explain, illustrate and demonstrate input and output of data using editing and text manipulation commands. Note: students should possess "touch" keyboarding skill. (Create sample text through keyboard or load test from file.) Observe as students practice manipulating text following specific instructions.	
	35% (P)	Have students use a problem-solving approach with word processing software to produce "mailable" output from rough draft copy or text creation exercises.	

MODULE 5: APPLICATIONS: COMPUTER SIMULATIONS

Status: Elective
Prerequisite: Module 1

Purpose

This simulation[1] module provides an opportunity to use the computer and its appropriate software in a formalized problem-solving application environment. Students are extensively involved in the various decision-making phases of the simulation. For some students, this project may likely be their first formal opportunity to become involved in detailed planning of all phases of a simulation.

Planning

Depending upon available resources, facilities, and goal(s), the teacher may take several approaches to this simulation. For example, students could proceed individually on separate projects, form groups and work as cohesive units on separate projects or one common project, or work as a class on the same simulation project. The type of simulation, the amount of detail, and the number of students involved will determine how this module is conducted.

In conjunction with the classroom teacher, the student(s) either identify one of the suggested topics listed in Module 5 of the *Computer Processing 10-20-30 Curriculum Guide* (pp. 27-28), or select a different topic suitable to both parties. Once a decision has been made as to how the module will be handled, the student(s) describe the specific problem, delineate the general thrust of the simulation, determine the scope, establish parameters under which the project will proceed, develop timelines and priorities, locate the necessary resources, identify hardware and software support and constraints, describe the anticipated outcomes, and present the proposal before obtaining final approval to proceed. This phase of the simulation requires that students do detailed planning, research, organizing, and communicating before they actually begin.

As the students progress through the various stages, the teacher should provide whatever assistance, support, or encouragement is needed. The time element will have to be monitored continually because students have considerable latitude in determining their rate of progress. Perhaps an activity timeline could be posted on the bulletin board and updated at identified intervals—a Gantt-type chart (horizontal bar chart showing timelines and progress schedules). Or, a critical path network technique such as PERT could be used.

---

1. Simulation may be defined as a technique which is an imitative representation of the functioning of a system or process and includes the use of a logical model (what the system does—data content and flow) and a physical model (how the job is done – personnel and processes). Powers, et al, Computer Information Systems Development: Analysis & Design (Southwestern Publishing Co., 1984).

Resources

In this project, students are encouraged to use the computer and its related software in a problem-solving application. Therefore, the first criteria in selecting a topic is the availability of appropriate hardware and software along with its appropriate documentation. Since additional resource information is entirely dependent upon the topic chosen, availability of resource support should be confirmed before a final decision is made.

The software used in this module may involve locally developed student programs, commercially purchased application software, or a combination of both. The identified objectives and specific emphasis placed on the simulation itself will determine the type and extent of software usage.

Evaluation

Teacher evaluation of this module potentially involves many factors. Students should be aware of the specific aspects of the simulation that are going to be evaluated and exactly how the mark distribution is going to be assigned before the module is started. (Some specific suggestions are listed in the "Teaching Suggestions" summary section of this manual.)

The "evaluation of the simulation" topic provides an excellent opportunity for students to assess the simulation experience realistically. This reflective exercise can be one of the most important elements in the module. Students can gain valuable insight into the evaluation process as they critically analyze and evaluate each phase of the project, regardless of whether or not the simulation was entirely successful.

If this module is effectively incorporated into a computer processing course, the potential benefits for both the students and the teacher are tremendous. Students will be challenged to demonstrate their initiative, resourcefulness, and academic and creative potential in using computers to solve a wide range of application problems.

Theoretical 30%	Practical 70%

COURSE: COMPUTER PROCESSING
MODULE 5 -- APPLICATIONS: COMPUTER SIMULATIONS

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
DEFINE SIMULATION Shelly – Chapter 10 (10.7-10.27) Sanders – Chapter 14 (367-368) – Chapter 16 (445-446)	25% (T)	Discuss what a simulation is. Present films depicting various types of simulation involving computers: – robotics – CAD-CAM – aerospace – flight simulation – weather forecasting – medical simulation – law enforcement	Evaluation based on: a) Simulation proposal b) Simulation project c) Evaluation of simulation d) Demonstrated software/hardware skills
(Resources are dependent upon the specific topic chosen.)		Explain the planning steps involved: – identify the problem(s) – set scope and parameters of the project – establish timeline and priorities – find resources – find hardware/software – determine intended outcomes Have students submit a written proposal (explaining the above items) for approval before proceeding with the project.	Suggested criteria: a) Amount and thoroughness of background research b) Originality/creativity c) Communicating ability d) Success of simulation e) Degree of difficulty of simulation f) Student assessment of the simulation g) Other factors?
Focus on topic(s) within subject area or Overview topics from several content areas or Develop an approved topic of particular interest	70% (P)	The simulation project could involve: – an individual student – a group of students – the entire class The simulation project could include: – one topic (from a content area listed) – several topics (in the curriculum guide) or – individually developed topic(s) from other area(s)	Marks awarded should reflect: a) The objective(s) of each phase b) Student involvement and participation – individual – group – class

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
EVALUATE THE SIMULATION	5% (T)	Have students: – compare their project to similar activities in the business world – evaluate the positive aspects of the projects – make concrete recommendations as to how the project could be improved or how changes to various procedures would affect the final outcomes – assess their personal experiences in completing this simulation Introduce and discuss each term as encountered in each topic. Present written or oral definition reviews. Basic terms should include at least: systems analyst, systems department, systems analysis, systems approach to problem solving, feasibility study, problem definition, systems design, systems implementation, systems evaluation, systems documentation, and feedback.	Oral and/or written report to the teacher or to the entire class. Assessment of the theory of the module should not count for more than 30%. Value of project should be at least 70%. The project should be evaluated in its entirety at the conclusion of the module with the theory and practical aspects incorporated into a final report. Using this approach, the final report would count for 100% of the student's grade.

MODULE 6: APPLICATIONS: DATA BASE

Status: Elective
Prerequisite: Module 1

Purpose

In this module, the student is introduced to the concepts of data storage and retrieval using a data base software system. The student also develops skill in the use of the software to solve application problems.

Planning

The introduction of a data base system should include a general discussion of the associated terminology, an explanation of what a data base is designed to do, a graphical illustration of how it works, and an identification of the ethical ramifications of what happens to the data and who has access to it.

The key concept is that an efficient data base system is designed to store interrelated data with a minimum of redundancy so that it can be effectively used for a wide variety of applications.

The introduction of students to the component parts of a data base would include such topics as the hierarchical organization of data (bits, bytes, characters, fields, records, files, and data bases), and the classification of data bases (types of information, purpose, goals and application).

If students understand the conceptual nature of how data bases function and how they can be used in the manipulation of data, they should be able to transfer that knowledge to other data systems outside of the school environment.

"Hands-on" application activities should include at least the following areas: the data base management system, data base creation, loading the data base, data base manipulation (searching, sorting, editing, and summarizing), data base catalog (querying and updating), output (data display and report writing), and data base design.

Resources

Students will need access to a data base software system with appropriate supporting documentation. It may be advisable to supplement the software-related instructions with a generic reference that conceptually explains how data bases are structured. If appropriate application problems are not provided with the support materials, teachers will need to develop suitable student problems.

Some teachers might prefer to deal specifically with applications that have some practical relevance to their classroom setting.

To aid student understanding of the data base framework, teachers may wish to use such aids as transparencies, pictures, or bulletin board displays to depict graphic examples of the hierarchical data base (tree structure), data base schemas and subschemas, and network data base or distributed data bases.

Evaluation

The measurement of student achievement in this module should reflect students' conceptual acquisition of information as well as the practical "hands-on" skill attainment. Theory tests could be used to determine the level of conceptual accomplishment. Practical "hands-on" tests involving either a demonstration of functional command knowledge or an actual data entry and/or information display report could be used to determine the level of skill attainment.

Theoretical	25%	Practical	75%

COURSE: COMPUTER PROCESSING
MODULE 6 -- APPLICATIONS: DATA BASE

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
MANAGEMENT OF INFORMATION Shelly – Chapter 9 (9.0-9.5) Sanders – Chapter 13 (344-352) – Chapter 14 (368-372, 406-414)	5% (T)	Conduct a class discussion of information flows (records management) and storage and retrieval methods (both manual and electronic). Identify problems associated with filing and finding information, security concerns, and ethical ramifications of confidential information. Appropriate magazine or newspaper articles dealing with various aspect of records or forms management and data base systems can be used as supplementary sources of current and interesting information.	1. Theory tests (25%) a) Objective – multiple choice – fill-in blank – matching b) Subjective – definitions – short answer – paragraph 2. Practical "hands-on" tests (75%) a) Demonstration of software/hardware manipulation b) Problem-solving applications using software
PARTS OF DATA BASES Shelly – Chapter 9 (9.1-9.8) Sanders – Chapter 2 (37-38, 40-47) – Chapter 12 (328-332)	5% (T)	Use blackboard sketches, transparencies, pictures, charts, and bulletin board displays to illustrate hierarchical component parts of data base. Establish the relationship of parts in the data base—character, field, record, file and data base. Solicit student contributions in class discussion.	
CLASSIFICATION OF DATA BASES Shelly – Chapter 9 (9.6-9.13) Sanders – Chapter 13 (344-356)	5% (T)	Identify types of data bases, their differences, and the advantages and disadvantages of each. Conduct a class discussion, prompting student involvement in supplying examples and illustrations. Graphically illustrate each type of data base, identify how they function, and discuss their similarities and differences.	
USE OF DATA BASE MANAGEMENT SOFTWARE Software Specification Documentation	65% (P)	Conceptually illustrate how the software functions. Demonstrate the actual steps involved in operating the software, including an explanation and demonstration of each specific command, and observe student operation and participation in application exercises. Activities should include: – file and record identification – file setup – data entry, editing, updating, deleting – file processing – report generation	

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
DATA INTEGRATION Shelly – Chapter 9 (9.6 - 9.10) Software Specific Documentation	5% (T)	Discuss how data and files can be transferred and/or integrated to update or merge records or files. Demonstrate actual operations if integrated software is available.	
DATA TRANSFER Shelly – Chapter 9 (9.11-9.13) Operating Systems Documentation Sanders – Chapter 10 (258-259) – Chapter 13 (347-349)	5% (T/P)	Illustrate and demonstrate practical uses for data transfer. Provide students with "hands-on" experience. Discuss communications aspects involved with on-line data bases. If appropriate, down load and up load information.	
TELECOMMUNICATIONS Shelly – Chapter 8 (8.1-8.35) Sanders – Chapter 10 (258-280)	5% (T/P)	Briefly discuss the various ways data can be transferred, the type of hardware used in data communications, and the role it plays in society. Demonstrate existing capability. Show films, pictures, or transparencies to help clarify student understanding.	
COMMERCIAL DATA BASES Sanders – Chapter 10 (258-263, 270-278)	5% (T/P)	Discuss commercial uses. Provide examples, illustrations, news stories and magazine articles, and solicit student participation. Supplement topic presentation with a film, a field trip, or a guest speaker. If available, access commercial data bases. Introduce and discuss each term as encountered in each topic. Basic terms should include at least: data information, data base, data base hierarchy, data base network, query language, distributed data base, data base schema, subschema, field, record, file, on-line communications, data transfer.	

MODULE 7: APPLICATIONS: ELECTRONIC SPREAD SHEET

Status: Elective
Prerequisite: Module 1

Purpose

In this module, students are introduced to the use of electronic spread sheets—how they work and what they are designed to do. This module also includes a conceptual understanding of how spread sheets function, and a practical knowledge of how to use the software to aid in a variety of problem-solving situations.

Planning

Students should be introduced to the basic design and general format of the spread sheet system, and allowed to develop a sufficient mastery of the specific commands required to operate the system efficiently in a problem-solving environment. It will thus be necessary to devote some time to an explanation of how the template is constructed, in addition to discussing the various coded commands and how they are intended to be used in manipulating the data. Once students are able to conceptualize how spread sheets function, they should be able to transfer that knowledge to other spread sheet systems that might be encountered outside of the regular classroom environment.

The introduction of this module would normally include a discussion of such topics as the computing environment (characteristics and capabilities of the specific software package in relation to the specific computer being used), the types of spread sheet applications (specific problem-solving applications using the work sheet), and an identification of system users (including individuals and organizations).

Ideally students should develop "hands-on" skill in at least the following areas: data entry (cell identification, labels, values, and formulas); editing/manipulation (inserting, moving, re-calculating, copying, deleting, and using built-in functions); files (loading and saving); and outputting (displaying and/or printing of work sheets from a variety of template sizes and layouts).

A number of approaches may be used to introduce students to spread sheet software. Sample spread sheet applications can be stored on disk beforehand. Then students load these problems into their computers and immediately begin using the various software commands to manipulate the stored data. Teachers may prefer to save only the template on disk and have students either enter their own data or enter supplied data via the computer keyboard. Teachers may wish to "walk through" the initial steps of entering simple spread sheet data with their entire class while the students execute each command as instructed.

Resources

Students will need to have access to a spread sheet software package and appropriate support documentation. Teachers may also wish to supplement the available reference materials with a generic reference.

Students should have an opportunity to work with a variety of application problems. If published exercises are not readily available, the teacher or students can design problem-solving examples that would be appropriate.

Evaluation

The evaluation process should reflect the time spent and degree of emphasis placed upon each topic. Practical application problem-solving questions could include students saving their completed test questions on disk and/or printing out the results in hard copy format.

Theoretical 25%	Practical 75%

COURSE: COMPUTER PROCESSING
MODULE 7 -- APPLICATIONS: ELECTRONIC SPREAD SHEET

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
MANAGEMENT OF INFORMATION	10% (T)	Conduct a class discussion about the management of information--why it is collected, how it is obtained, who receives the information, how it is used, and what happens to it afterward. Encourage student contributions through discussions, reports, or bulletin board displays involving such topics as access to information legislation, security, or confidentiality. Invite guest speakers who are knowledgeable about the topic(s).	1. Theory tests (25%) a) Objective – multiple choice – fill-in blank – matching b) Subjective – definitions – short answer – paragraph 2. Practical "hands-on" tests (75%) a) Demonstration of software/hardware manipulation b) Problem-solving applications using software packages
USE OF SPREAD SHEETS Sanders – Chapter 4 (82) – Chapter 8 (219, 222, 223) – Chapter 14 (365-366)	40% (T/P)	Compare similarities and differences of manual and electronic spread sheets by using films, transparencies, charts, or actual hard copy printouts. Identify, illustrate, explain and demonstrate specific operational commands by using computers and appropriate software, transparency examples, or summary command handout sheets. Observe student operation, skill development on keyboard technique, command familiarity, and application fluency. Ideally students will develop functional skill on: template development, data entry, editing, processing, files, and output.	
SPREAD SHEET APPLICATIONS	50% (T/P)	Discuss spread sheet applications in a wide variety of settings: home, business, industry, research, etc. Have students create their own templates or develop them from pre-formatted models. Provide students with an opportunity to experiment with and use a variety of commands in their problem-solving spread sheet activities. The application activities could involve a series of minor problems from a variety of interest areas, or a major simulation approach to problem solving.	

MODULE 8: OVERVIEW OF INTRODUCTORY PROGRAMMING LANGUAGE

Status: Elective
Prerequisite: Module 1

Purpose

This module attempts to bridge the rather broad range of backgrounds in an average classroom of students. Some students may have been introduced to many of the concepts of this module in previous courses or through private study.

Throughout this module, as well as all others in the computer processing stream, emphasis should be placed on problem solving. The motivational value of computer programming in teaching problem-solving techniques should not be underestimated. Problem solving can be handled starting with the organization of the problem for solution by computer, through debugging strategies, and then testing strategies.

The emphasis placed on professional standards of programming in this module will depend upon what other programming modules, if any, are used in a course. If this is the only programming module to be used, these concepts may be de-emphasized, but if a course is to be designed using primarily programming modules and structured problem solving, documentation standards and other professional aspects should be emphasized from the start.

This module is either directly or indirectly prerequisite to Modules 9 through 16, and Module 18.

Planning

Introduce computer programming by comparing computer language to natural language.

Individualized materials work well for the programming aspects of this module (as well as the other programming modules). They allow students who have a strong background to move through the material more quickly. Enrichment activities should be available for those students who finish the "required" material. This material need not be highly structured; perhaps it could allow study in an area of personal interest. Topics could include use of software packages and languages not studied elsewhere in the course, or the use of hardware which may be demonstrated in the course but not normally included in detail.

Resources

Software packages that introduce the hardware and basic operating procedures are available for most equipment. These packages are either standard with the equipment, or can be purchased at reasonable cost. Commercial software packages that teach introductory programming and cover topics useful for enrichment are also available.

MECC has a program that adds a number of significant structured programming instructions to Applesoft BASIC that are already present in some other versions of BASIC (e.g., IF/THEN/ELSE, WHILE, REPEAT/UNTIL, CASE plus some other additions). This 'modified' BASIC could be used from the start or could be introduced in one of the later modules.

It is suggested that for this module there be a ratio of one computer per student. If students are required to work together, carefully select partners and ensure that both partners have equal access to the keyboard.

If Apple computers are being used, an editor program may be considered. The editor program could be used from the students' first contact with programming, or introduced after the students have become accustomed to unenhanced editing on an Apple.

Evaluation

Approximately 80% of the final grade should be based on practical work, which would include primarily programming assignments. The weighting of various aspects of programming assignments in this and other modules is summarized in Appendix F.

Theoretical	20%	Practical	80%

COURSE: COMPUTER PROCESSING
MODULE: 8 – – OVERVIEW OF INTRODUCTORY PROGRAMMING LANGUAGE

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
TERMINOLOGY Shelly – Chapters 1-3 McRitchie – Chapters 1-6 (selected topics) Mandell – Chapter 1 (2-22)	1-5% (T)	Study the vocabulary as it arises in the discussion of the topics. Terms to be covered should include: computer, input, output, data, processor unit, central processing unit (CPU), main computer storage, minicomputer, microcomputer, mainframe computer, stored program, solid state, software, system software, microelectronics, microprocessor, CAI.	Emphasize the ability to use terminology rather than recite textbook definition. Fill-in-the-blank and matching questions could be effective.
	≤	Use a program to create crossword puzzles after introducing the terminology.	
HARDWARE Shelly – Chapter 1 (1.1-1.13, 1.19-1.28) – Chapter 3 (3.1 - 3.23) Mandell – Chapter 1 (2-22) – Appendix H (408)	5-10% (T/P)	Use Appendix C to put hardware into the overall perspective of a computer system. Special "programming" keys can be introduced through the use of commercial programs designed for that purpose. A purchase specification for a computer system written by the students can be a good exercise in applying vocabulary and knowledge of hardware correctly. This could be used for evaluation as well.	A "lab type" examination with a computer system that the students have not used can be effective. Tie a piece of string around the various hardware components, number them, and have the students identify them generically. This approach will be especially effective if the "test" system is physically arranged very differently from the system the students use, perhaps even being a Mini instead of a Micro.
SOFTWARE Sanders – Chapter 20 (542-583) Mandell – Appendix I (409-413) Clark & Drum – Chapter 1 (1-14)	10% (P)	DOS and Operating System instructions should be the emphasis here. These instructions can be handled by preparing class sets of diskettes with one or more of each type of file (e.g., text, Applesoft, Integer, binary), and having a predefined set of operations to be performed on that diskette. (See Appendix A for example.)	
		Appendix C can be used to put DOS and Operating System instructions in perspective relative to a computer system. Discuss software that is in hardware (e.g., BASIC in ROM), and use the term "firmware" to describe it. Emphasize that computer media (e.g., a diskette) is not software, but is simply a means of recording, storing, and transporting software. (A book is not knowledge, it is only the media used to record, store and transmit knowledge.)	Students can be given a series of commands similar to Appendix A, and asked what each step does and what the final catalogue of the disk would look like.
			Observe the students' use of the DOS and operating system instructions to assess their understanding.

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
PROGRAMMING TECHNIQUES McRitchie – Chapters 1-6 Braun – Projects 1, 2 Sanders – Chapter 21 (584-624) Mandell – Chapters 2-7 (25-151) Clark & Drum – Chapters 1-7 (1-112) – Chapter 9 (133-150) – Appendix B (270-271) – Appendix D (276-289)	50% (P)	Students should assess their programs DURING the program development process itself, rather than as an afterthought. At least some familiarity with the concepts of structured problem solving would be helpful before the students begin learning about programming. Self-paced materials work best here to allow for students' previous knowledge and experience. These students often overestimate their proficiency, and therefore should be required to complete the materials. When setting up self-paced materials, try to strike a balance between presenting material fast enough to maintain student interest, but slow enough for novice programmers. Areas of special interest can be used as enrichment for those students who have experience and will be able to complete the material quickly. For example, work with specialized hardware like modem, graphic tablet, mark sense reader, and plotter, or with specialized software packages not used in the course. When dealing with the concept of variables and data storage, be sure to cover the concepts of integer and string variables as well as real variables. Students may be provided with a "Buggy" program which they are to correct. Bugs come in two varieties: syntax errors (a violation of the rules of the language), and logical errors (errors in the logic or order of operations). The former are much easier to find because the computer stops; the latter are harder to find and must be guarded against more carefully by planning. Logical errors often manifest themselves in a program at some other point than where the error actually occurs. In order to emphasize the computing science concepts rather than programming in one particular language, discuss each language instruction within the general framework of the "primitives" of all computer languages: Input, Output, Assignment, Decision/Selection, Repetition. (See Appendix E.)	Specific questions about the syntax and effect of commands can be used as well as a description of what a sample program does by following its execution manually. Students can do questions from the end of chapters in the text.

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
PROGRAMMING APPLICATIONS Mandell – Chapter 8 (153-170) Clark & Drum – Chapters 1-9 (1-150)	30% (P/T)	Appendix F has a suggestion regarding documentation and evaluation standards for programs. Discuss the importance of thoroughly testing programs and discuss what constitutes good test data. Test top and bottom margin values by using barely valid quantities and barely invalid quantities, as well as typical values For example, a program that asks for student percent marks could use test values of 100, 100.1, 0, -.1 to test margin values. Emphasize the importance of making sure that every single routine in a program is tested in this manner.	Programming assignments can be given in two versions. The first version can be relatively easy, covering the intended topics. The second version can add additional features and provide challenge. The first version can be worth the majority of the marks, with the remainder of the marks being given for the "enhancements". (See Appendix D.) At this introductory level, elegant and sophisticated algorithms can take a back seat to the correct use of programming techniques and having the program work correctly. While other students are continuing with the self-paced material, go over each student's program individually. Make sure that each student understands the reasons for the mark. 1. Theory (20%) a) Objective – multiple choice – fill-in blank – matching b) Subjective – short answer – definition – paragraph

MODULE 9: FUNDAMENTALS OF INPUT/OUTPUT

Status: Elective
Prerequisites: Modules 1 and 8

Purpose

An ever-increasing degree of desirable programming style should be developed in this module. A five-credit course that is designed to provide an introduction to programming could include only Modules 8 and 9.

Students often relate INPUT to a keyboard and OUTPUT to a screen or printer. This module teaches the wide variety of methods and devices that can be used to perform these tasks.

Planning

Although this module could be taught with a machine/student ratio of 1:2, a ratio of 1:1 is much preferable so that students can develop their own programming styles and practise independent problem-solving skills.

Resources

Attached as Appendix G is a text and graphic format sheet which can be reproduced and distributed to students for use in planning computer output, and will be useful for other modules as well.

Evaluation

Theoretical	25%	Practical	75%

COURSE: COMPUTER PROCESSING
MODULE 9 -- FUNDAMENTALS OF INPUT/OUTPUT

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
INPUT/OUTPUT PROCESSES Shelly – Chapter 5 (5.1-5.35) – Chapter 6 (6.1-6.25) McRitchie – Chapter 3 Sanders – Chapter 6 (134-162) – Chapter 7 (185-199) Mandell – Chapter 1 (9, 11, 15, 16) Clark & LaBarre – Chapters 1-6* (selected topics)	10% (T)	Introduce and discuss each term as encountered: keypunch, transactional and batch oriented systems, interactive, optical mark reader, optical character recognition, bar code, universal product code, magnetic ink character recognition, and point of sales terminal. Discuss industrial I/O devices such as pressure transducers, temperature sensors, servo motors and robotics. The method of representing data will often depend on how that data will be input to a computer system, or how it will be output (e.g., the use of UPC as a method of representing data for grocery items because of input method). Emphasize the CHARACTERISTICS of Input and Output devices rather than exhaustive lists of current devices. Distinguish between the Input and Output DEVICE and the INTERFACE. The device is the part of the hardware that accepts or presents information from the outside or to the inside of the computer system, often (but not always) from the "physical" world rather than the "electronic". The interface converts the computer's representation to one that the output device can use, or converts the input into a form that the computer system can store and use. Remind students about input devices like pressure transducers, thermocouples, pH meters, etc., and output devices like motors. Students can make bulletin board displays of sample computer input such as universal product codes. Arrange field trips to facilities that use various types of input/output devices (e.g., grocery store, department store). (Some of this material has been covered with a different emphasis in Module 3.)	Evaluate students' ability to recognize the characteristics of Input and Output devices. Have students categorize some devices which have not specifically been discussed in class.
INPUT PROCESSES	5% (P)	Draw attention to the fact that the input of data is often the last step before the information becomes part of the "data base". Although this may seem to be a relatively unimportant aspect of data processing, the majority of "computer errors"can be traced to carelessness in data input. (This topic can be integrated with some of the topics of Module 3.)	Present a number of "computer errors" to the students and ask them to assess the likelihood of a hardware error, software error, or data entry error.

*Specific instructions to BASIC - do not deal with generic concepts.

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
INPUT PROCESSES (cont'd)		Discuss the distinction between batch and transaction based systems as it relates to methods of entry terminology: GIGO (Garbage In, Garbage Out), local and remote job entry.	Examples hardware: a document is printed on a word processor but the letter "A" never prints; software: a person receives 53 copies of a magazine; data entry: a person receives a tax refund check for $143.23 instead of for the correct amount of $134.23.
	10% (P)	Output requirements will be quite different depending upon the specific output device (e.g., a monitor may be 40 characters wide and 24 lines deep while a printer page may be 80 characters wide and 66 lines deep).	
PROCESSING OF DATA Shelly – Chapter 4 (4.1-4.20) Sanders – Chapter 5 (99-122)		Describe the manner in which the CPU manipulates and stores data.	
ARRAYS Shelly – Chapter 1 (4-6) – Pages 7, 8, 9 – Appendix A (A.30-A.36) Sanders – Chapter 21 (585-618) Mandell – Chapter 11 (213-232) – Chapter 13 (253-283) Clark & Drum – Chapter 10 (151-168) – Chapter 11 (169-186) Clark & LaBarre – Chapter 3 (72-87) – Chapter 7 (195-226)	5% (P)	Have students write programs incorporating single and double subscripted variables. Demonstrate the use of table handling and searching techniques to manipulate data and to solve problems. Single subscripted variables can be equated to the room numbers in a single-storey building: building name = array name, and room number = element within the array. Double subscripted variables can be equated to a multi-storey building: building name = array name, floor number = first index of array, and room number on that floor = second index to array.	

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
LIBRARY AND OPERATING SYSTEM FUNCTIONS Shelly – Appendix A (A.29-A.41) – Workbook Appendix A (A.1-A.8) Sanders – Chapter 21 (590-595) Mandell – Chapter 10 (200-211) Clark & Drum – Chapter 12 (187-214) Clark & LaBarre – Chapter 3 (72, 92-93)	5% (P)	Emphasize functions that students would most likely encounter in a business environment (e.g., RND, INT, MID$, LEFT$, RIGHT$, LEN, CHR$). Introduce a more general formula for generating random integers between two values: X = INT (UL-LL*RND(1) + LL), where UL = UPPER LIMIT and LL = LOWER LIMIT.	
PROGRAMMING APPLICATIONS Shelly – Appendix A (A.37, A.41-A.42) – Workbook Appendix A (A.9-A.10) Sanders – Chapter 18 (477-510) Mandell – Chapter 9 (196-198) – Chapter 10 (210-211) – Chapter 11 (228-232) Clark & Drum – All chapters (Programs to Write) – Appendix B (270-271) – Appendix D (276-289) Clark & LaBarre – All chapters (Programs to Write) – Appendix C (433-445)	60% (P)	Have students write programs to solve various problems: design, flowchart, code, test, debug, run.	See Appendix F for suggested weighting for programming projects for this module.

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
DATA COMMUNICATIONS Shelly – Chapter 8 (8.1-8.33) Sanders – Chapter 10 (259-280)	5% (T/P)	Introduce and discuss each term as encountered: modem, terminal, acoustic coupler, coaxial cable, fiber optics, communications control unit, front-end processor, line speed, baud rate, synchronous, asynchronous, simplex, half-duplex, full-duplex, point-to-point and multi-drop lines, local area network, ring and star configuration, multi-programming, intelligent and dumb terminal. Stress the importance of data communications to business applications of computers. Discuss the increasing importance of local area networks (LANs). Distance Signal Travels → Type of Communication up to about 10m ------------→ within a single computer system 10m – about 1km 1 room/bldg/campus ------------→ local area network over 1km to 1000km city/country ------------→ long haul over 1000km ------------→ interconnect of long hauls If you have direct phone access to any of the metropolitan areas in Alberta and have a modem, you may be able to gain access to free bulletin boards which can be used to demonstrate data communications.	

MODULE 10: INTRODUCTION TO ADVANCED COMPUTER PROGRAMMING TECHNIQUES

Status: Elective
Prerequisites: Modules 1 and 9

Purpose

Courses using this module and Module 11 should be aimed at groups of students who want to develop advanced problem solving techniques in a programming framework.

Students will synthesize solutions to more complex programming problems to develop their problem-solving skills.

Planning

At this stage students should have a good understanding of structuring techniques. In programming assignments, the value and importance of structuring solutions and making programs modifiable can be emphasized by making minor changes to the program specifications part way through the assignment. For example, a program that repeats something a fixed number of times can be modified so that the number of repetitions is input from the keyboard. These kinds of changes often occur in a programming environment, and are relatively easy to make if the program is well-structured.

Evaluation

An ever increasing emphasis on problem-solving strategies should be given in this module and the next module. This is reflected in the increased emphasis on organizational aspects of programs as suggested in Appendix F. The students' systematic approach to problem solving will be reflected in evaluation of activities.

Theoretical 20%	Practical 80%

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
NUMBERING SYSTEMS Shelly – Appendix C (C.1-C.9) McRitchie – Chapter 12 (585-588) Sanders – Chapter 5 (107-110)	20% (T)	Introduce and discuss each term as encountered: binary, bit, byte, decimal, hexadecimal. The study of number systems is important to a skilled programmer in terms of understanding the mode of operation of a computer system. The computer stores and manipulates data as a sequence of voltages/no voltages which are represented by programmers and electronics people as 1's and 0's -- these are the digits of the binary number system. The concept of hexadecimal is important because it is easier to work with than binary, but can be quickly converted to binary. Describe decimal, binary and hexadecimal as positional numbering systems. Discussion of them in this light simplifies the concepts. As a challenge question, ask students to explain the following number sequence in light of their knowledge of number systems: 10, 11, 12, 13, 14, 15, 16, 17, 20, 22, 24, 31, 100, 121, 10000. The answer is that these are representations of the magnitude 16 base ten in number systems with base 16, 15, 14, 13, 12, 11, 10 . . . 2. Number systems are only different ways of representing mathematical magnitudes. For example, 7 and VII are representations of the same magnitude using Arabic and Roman numerals. Give students the hexadecimal addresses of some visually dramatic flag locations. Have them convert to decimal and use poke of the decimal address to set the flag. Note: Have the students remove their disks from their disk drive and do a cold reboot afterwards to avoid damaging data on their disks.	Distribute a work sheet that provides a value in one number system and requires that students convert to the other two (decimal, hexadecimal and binary). In addition to conversion to the three number systems taught, have the students convert to a non-standard number base like base 5. Ask students to explain the difference between positional and non-positional number systems (e.g., decimal and Roman numeral).

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
"PROFESSIONAL" PROGRAMMING TECHNIQUES Shelly – Chapter 12 Mandell – Chapter 2 (25-40) – Chapter 8 (153-170) Clark & LaBarre – Chapter 1 (2-40) – Chapter 4 (118) – Chapter 6 (183-191)	20% (P)	See Appendix B for some specific guidelines to emphasize the topic of "professional" programming.	All programs that are marked by the teacher should demonstrate the students' understanding of "good programming" techniques. See Appendix B for characteristics of good programs and Appendix F for a suggested marking guide and weightings of programming components.
SUBROUTINES Shelly – Chapter 11 McRitchie – Chapter 11 (508-524) Mandell – Chapter 9 (171-198) – Chapter 10 (199-211) Clark & LaBarre – Chapter 1 (5-13) – Chapter 3 (70-103)	20% (P)	Emphasize the use of subroutines in modularizing programs, one module (one task), and one subroutine. Have students start subroutines with a comment briefly naming and describing the subroutine, and have them use whole thousand line numbers for subroutine starts.	When marking students' programs stress the wise use of subroutines for modularization.

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
MULTIPLE DIMENSION ARRAYS Braun – Projects 7-10 McRitchie – Chapter 8 Mandell – Chapter 11 (213-232) Clark & LaBarre – Chapter 3 (85-87) Sanders – Chapter 21 (606-610)	20% (P)	The concept of multi-dimension arrays can be introduced by using the analogy of a 1-dimension array being a list of numbers on a page, a 2-dimension array being a table of numbers on a page, a 3-dimension array being a number of pages of tables of numbers, a 4-dimension array being sets of books of pages of tables of numbers, etc. Point out to students the amount of memory used by multi-dimension arrays. For example, a school of 80 teachers each with 7 classes of 30 students could not store their marks with DIM MA (30, 7, 80), because 30 x 7 x 80 elements x 5 bytes per element for a Real array = 84000 bytes. This is more memory than most micros have, not even allowing for program and system use. An integer array at 2 bytes per element would use only 33600 bytes, which is why multi-dimension arrays are often integer when possible. Similar calculations can also be done for other memory sizes.	
INTRODUCTION TO GRAPHICS (User manual specific to computer.) Clark & LaBarre – Chapter 5 (145-165) – Chapter 12 (353-371) – Chapter 13 (372-402)	20% (P)	Number systems can be tied in here through use of shape tables if you are using Apple computers. Give only an elementary treatment of graphics; a more in-depth treatment is possible through the use of Module 16. Check the content of that module before proceeding with this topic to avoid overlap.	

MODULE 11: ADVANCED COMPUTER PROGRAMMING TECHNIQUES

Status: Elective
Prerequisites: Modules 1 and 10

Purpose

Students will find this module on the study of files and file manipulation techniques challenging. They will have to use advanced problem-solving skills because the creation and manipulation of files is often not reflected directly in any output on the screen.

Students will gain an appreciation of the importance of files as storage structures for data in a business environment. The concept of files and records that underlie the specific operating procedures for a particular computer system should be stressed.

Planning

In assisting students with debugging programs that use files, resist the temptation to correct their problems. Rather, assist them in analyzing what the problem is, where it occurs, and how to solve it. In this way abstract analytical skills can be developed.

A filing cabinet analogy can be effective in introducing computer files. A filing cabinet is like a disk file. (Or is a disk file like a filing cabinet?) Identification of some type will be on the outside of the cabinet indicating the contents of the cabinet (customer files, 1985). Inside the cabinet (file) will be a filing jacket (record) for each customer (Joe Blow). Each filing jacket (record) will contain individual entries about that jacket's namesake (name, address, etc.).

Resources

A commercial data base program should be available to demonstrate and give students a conceptual framework for the study of this module.

Evaluation

Theoretical	20%	Practical	80%

COURSE: COMPUTER PROCESSING
MODULE 11 -- ADVANCED COMPUTER PROGRAMMING TECHNIQUES

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
TYPES OF FILES Shelly – Chapter 7 (7.1-7.22) Braun – Project 5 Sanders – Chapter 7 (175-184) Mandell – Chapter 12 (233-252) – Chapter 14 (285-306) – Chapter 15 (307-328) Clark & Drum – Chapter 13 (215-236) – Chapter 14 (237-257) Clark & LaBarre – Chapter 4 (104-143) – Chapter 9 (249-290) – Chapter 10 (291-325) – Chapter 11 (326-352)	10% (T)	Introduce and discuss each term as encountered: – auxiliary storage, parity, tape, hard disk, floppy disk, Winchester technology disk, track, sector, file, record, random access, sequential access, indexed file Discuss how DATA files fit in with the general scheme of files. FILES Program Source Object Data Include a discussion of various popular auxiliary storage media: Magnetic Disk Hard fixed removable Winchester Technology Floppy mini 8-inch Tape Digital Audio Physical Punched Card (Hollerith) Paper Tape Include a relatively detailed explanation of the particular type of auxiliary storage and details of track/sectoring being used by the students. This topic should include types and methods of accessing files, and types of auxiliary storage devices. Introduce a conceptual framework for files: – disks are divided into files: customer file, supplier file – files are divided into records: customer #47, Joe Blow – records are divided into fields or entries: Joe Blow 123 Main St. Anyplace, AB T0K 0Y0	

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
FILE HANDLING PROCEDURES (User manual specific to computer.) Braun – Projects 5, 7, 8 and 9 Mandell – Chapter 14 (305-306) Clark & LaBarre – Chapters 4, 9, 10, 11 (Programs to Write)	10% (T)	Introduce and discuss each term as encountered: detail report, summary report, exception report. Develop a conceptual framework for the usefulness of files through manipulation of a commercial data base program (by student or teacher). Why use data files on disk or tape? (For long-term storage of information and to store more information than will fit in the computer's main memory.) The type of file selected will depend upon the manipulation that must be performed on the file.	
	80% (P)	Set up a project: – Have the students create a file. – Develop a program to convert the data structure (or organization of field) to another structure. – Create a third program to manipulate this new structure.	

MODULE 12: EXTENDED PROGRAMMING PROJECT (See also Module 15)

Status: Elective
Prerequisites: Modules 1 and 8

Purpose

In this module, students use and integrate the knowledge acquired in the other programming modules of the course. Most or all of the programming projects to this point have been defined by the instructor. This module requires the student to do the problem definition as well as the other aspects of the program.

The emphasis should be on the development of a single relatively complex program. (Note that Module 17 emphasizes the investigation, specification, creation and implementation of a SYSTEM.)

Planning

Projects can come from within the school. The instructor can make preliminary contacts with other teachers in advance, explaining what the students will require. Teachers may be able to specify school or non-school related problems. The students themselves, however, should make the contact with "customers" and discuss the program specifications with them. Potential customers should be reminded that the resulting program is 25 hours of student work and that the students are DEVELOPING the skills of programmers; therefore the resulting product may not be comparable to a commercially produced program. Students should also be encouraged to assess carefully how extensive a project they will be able to undertake in the specified time.

Many of the topics in other modules relate well to the kinds of tasks the students will be performing in this module. Students could be encouraged as well to undertake projects of their own specification.

Although the problem definition phase of the extended project may at first seem relatively simple for student-initiated projects, it will often become clear that this phase of the project requires a considerable effort. The students' tendency to try to skip this phase because they "know" what they want to do may lead them into trouble.

Because this module is very much oriented to independent activity, take care to have the students develop and follow a timeline. Also take care that the projects are not too extensive for the time frame allowed. If suitable projects are found that will require too much time to complete, they may be better handled in conjunction with Module 17.

Evaluation

If each student is developing a different program, the students can be required to keep a daily diary which the teacher can check each day. This diary and the actual program can be used in the evaluation procedures. Students can also give oral presentations on their projects to the class.

Evaluation should emphasize the preliminary planning involved before the production of any programs, and project follow-up and evaluation after implementation of the program.

It is important that the students understand the evaluation scheme for this module before they start. The emphasis that will be placed on planning and organizational skills should be clear.

Students could be required to produce a run-time or user manual for their program as well, explaining hardware, operating system, peripheral requirements, detailed instructions on using the program, error messages that may be encountered and corrective action required, etc.

Theoretical	15%	Practical	85%

COURSE: COMPUTER PROCESSING
MODULE 12 -- EXTENDED PROGRAMMING PROJECT

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
DEVELOP A PLAN Sanders – Chapter 18 (479-504)	10% (T/P)	Have the students establish a timeline for various aspects of the project as part of the initial planning stage. (Give consideration to using a GANTT chart or PERT planning approach.) Document both the program and the students' activities (e.g., daily activity log or diary).	Assess the thoroughness of the students' definition of the problem BEFORE starting to create the program. The amount of additional information and changes in information between the initial and the final problem definition can be an indication of the thoroughness and care taken in planning. Large differences may reflect poor initial planning.
PREPARE A PROGRAM Sanders – Chapter 19 (512-536) Clark & LaBarre – Chapter 1 (2-41)	75% (P)	Have students develop a program that reflects basic "user friendly" criteria.	
HARDWARE AND SOFTWARE REQUIREMENTS	5% (T/P)	Have students indicate minimum and optimum requirements for the memory, operating system, storage media, printer, etc.	Evaluating the documentation of the program itself covers only one aspect. If a daily diary is used, the effectiveness of this type of documentation can also be used in determining students' performance.
EVALUATE THE PROGRAM	10% (T/P)	Have students do a critical analysis of their performance in the project. – How well they estimated and followed their timelines. – How well they determined requirements for the program. – Identify problems encountered in the project, and indicate what could have been done to minimize these problems.	Another criteria for evaluation might be the students' ability to set up accurate timelines for the project. Look for evidence of ongoing internal and external documentation throughout the project, rather than documentation as an afterthought. Assess the students' own evaluation of the project.

MODULE 13: INTRODUCTION TO SECOND PROGRAMMING LANGUAGE

Status: Elective
Prerequisites: Modules 1 and 8

Purpose

In this module, the student is introduced to another high-level programming language. Emphasize the recognition of differences AND similarities in high-level languages, rather than syntactic descriptions of the languages. Also, emphasize the advantages of one programming language over another for particular applications. There is no "best" computer language for all purposes.

Planning

Choosing the second high-level language to learn will depend on a number of criteria. What alternative languages are available for the equipment and the configuration that is available? And what was the first language learned? If the first language learned was an interpreted language, then a compiled language would be a good second choice. If the first language was relatively unstructured, then a more structured language would be a good choice. The second language can be nicely tied in with the first language through study of the primitives of all programming languages (Appendix E). This framework provides a rationale for the study of a second language by pointing out the similarities and differences among various languages.

Resources

Specific programmers' manuals for the language chosen should be available to students as references.

If Apple Pascal is chosen, MECC has a set of sample Pascal programs that demonstrate the use of Pascal instructions (MECC #748 Manual and Diskette), and a Pascal User's Guide (MECC #627 Manual).

Evaluation

In programming assignments, watch for efficient use of the instruction set of the new language. Often students will try to continue programming as if they were using a language with which they are more familiar.

Theoretical 35%	Practical 65%

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
COMPARISON OF LANGUAGES Shelly – Chapters 12.1-12.6 Sanders – Chapter 19 (517-532) Mandell – Chapter 1 (16-20)	5% (T)	Discuss how computer languages fit in the scheme of things, how the various high-level languages are related to one another, and how they differ (Appendix C). Specific languages are generally designed for specific purposes: FORTRAN for formula translation, COBOL for business and file manipulation, LOGO for education and graphics, PASCAL for teaching structured programming, etc.	Short-answer questions regarding the advantages and disadvantages of one class of languages over another, e.g., compiled versus interpreted. Have students specify which class of computer language would be best for suggested applications.
HARDWARE/FIRMWARE SPECIFICATIONS	<5% (T)	Discuss any specific requirements for the language the students are going to use, such as special cards or chips, amount of memory, etc. If the language to be used has no special requirements, discuss the language's use of memory as compared to other language(s) studied.	
LANGUAGE SYNTAX Sanders – Chapter 20 (542-583) – Chapter 21 (584-624) Mandell – Chapters 2-11 (25-232) Appendices Section Clark & Drum – Chapters 1-11 (1-186) Appendices Section Clark & LaBarre – Chapters 1-8 (2-248) Appendices Section	50-55% (T/P)	Discuss each of the new instructions in terms of the primitives of all computer languages. (See Appendix E.) Introduce various methods for the specification of computer language syntax -- these might include Baucus-Naur and Train diagrams. It is advantageous to use a generic method of defining syntax so that students can determine for themselves what uses of commands are valid or invalid. MECC has a set of sample UCSD Pascal programs (#748 Manual and diskette) that can be used as sample programs for students to type in, compile and run. It is not advisable to give the students the source files themselves; have them type in their own programs. Mention some of the advanced features of the language being studied, but study in detail only the basic input, output, assignment, decision and repetition instructions (because of time constraints).	Observing the students' understanding of the use of language as they develop programs is the best way to spot students who are having difficulty. To test students' understanding of the use of syntax descriptions, give a syntax description of a nonsense instruction, and have the students evaluate the validity of several uses of the instruction.

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
APPLICATIONS OF THE LANGUAGE Mandell – Chapters 2-11 ("Programming Problems") – end of each chapter Clark & Drum – Chapters 1-11 ("Programs to Write") – end of each chapter Clark & LaBarre – Chapters 1-8 ("Programs to Write") – end of each chapter	25% (P)	Emphasize specific advantages of the language being used for specific applications.	Ask students to recommend a language of their choice for various suggested applications; e.g., – File Manipulation: COBOL – Statistics: FORTRAN – Recreational Programming: BASIC, etc.
MODIFICATION OF PROGRAMS	5% (P)	Ask students to make modifications to the MECC programs mentioned above, and to debug these programs.	Assess students' approach to debugging as they work on programs. Is there a debugging strategy? Can the students determine the general area of the program where the error occurs and the type of error they are looking for?
	5% (P)	Introduce bugs into the MECC programs and have students debug them.	
OUTPUT	5% (P)	Have students demonstrate basic formatting skills in order to display and/or provide hard copy of the output.	

MODULE 14: APPLICATIONS IN SECOND PROGRAMMING LANGUAGE

Status: Elective
Prerequisites: Modules 1 and 13

Purpose

This module allows for the further study of a second high-level language. The emphasis should be on specific characteristics of the second language that distinguish it from other high-level languages—not just syntactically, but conceptually.

Practical problem-solving in a programming environment is also an important aspect of this module.

Planning

The most effective projects for achieving the goals of the module are applications which would be difficult or cumbersome to implement in the other language studied.

Resources

The second high-level language must be chosen as the basis of what is available on the computers in the classroom. Programming reference manuals should also be available for the students' use.

Evaluation

Theoretical	25%	Practical	75%

COURSE: COMPUTER PROCESSING
MODULE 14 -- APPLICATIONS IN SECOND PROGRAMMING LANGUAGE

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
APPLICATION OF LANGUAGE*	20% (T/P)	Explore the specialized and unique features of the language. If the language is very complex or has many subtleties, more advanced features may be omitted.	
MODIFY AND DEBUG PROGRAMS	50% (P)	Have students modify existing programs, such as the MECC program mentioned in Module 13.	
LANGUAGE SYNTAX	10% (P/T)	Mention, if applicable, that some business languages have particular ways of representing syntax (e.g., Train diagram for Pascal). These are usually shown in reference manuals.	
PROGRAM DOCUMENTATION	10% (P)	Discuss "suitable" documentation techniques -- use of indention, use of long and descriptive variable names, etc. (Refer to Appendix B.)	Look for appropriate use of indention, variable names, etc., if appropriate.
FORMAT OUTPUT	10% (P)	Draw attention to any special formatting capabilities of the language.	

* Continue using the same text(s) as in Module 13.

MODULE 15: EXTENDED PROJECT IN SECOND PROGRAMMING LANGUAGE

Status: Elective
Prerequisites: Modules 1 and 13
(See Module 12 for Teaching Suggestions and Evaluation Approaches)

Purpose

In this module, students apply their knowledge of a second high-level programming language by developing some relatively advanced programs. Specific suggestions in Module 12 are also applicable to this module.

Planning

See Module 12, which is very closely related.

Students' projects should be relatively complex, realistic, and appropriate to the nature of the language.

Evaluation

Theoretical 15%	Practical 85%

MODULE 16: GRAPHICS

Status: Elective
Prerequisite: Module 1

Purpose

Module 16 has high motivational value for students; it can be used to advantage in refining programming and problem-solving skills. Students should gain an appreciation of how the graphic representation of data can aid in the presentation of information. Projects could include the use of various types of graphs of numeric data, or graphics as part of a simulation.

Planning

Projects and examples which use business or simulation graphics are most suitable for this module. Shape tables, block graphics, sprite graphics, and turtle graphics can also be discussed, demonstrated, or used.

Implementation and emphasis will be very much affected by the graphic capabilities of the computer system being used.

Resources

The computer must be able to handle at least some graphics programming. It is not necessary, however, for the system to handle all of the specific aspects mentioned or all of the suggested methods of dealing with graphics. Some of them can be covered as theory.

Specific reference and programming manuals for the system being used should be available.

Appendix G is a text and graphics layout planning sheet which can be reproduced and distributed to students.

Tours of CAD/CAM (Computer Aided Design/Computer Aided Manufacturing) facilities will emphasize the value and usefulness of graphics in business and the world of work.

Evaluation

Evaluation criteria should include the effectiveness of the graphics used to represent data.

Theoretical 35%	Practical 65%

COURSE: COMPUTER PROCESSING
MODULE 16 -- GRAPHICS

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
PROGRAMMING APPLICATIONS	5% (P)	Explain that graphics programming is simply programming that uses a different form of output.	
Sanders – Chapter 7 (195-198)			
GRAPHICS MODES	50% (P/T)	Commands will be very specific to the system being used.	
Clark & LaBarre – Chapter 5 (145-169) – Chapter 12 (353-371) – Chapter 13 (372-402)	20% (T/P)	Demonstrate and have students use the restriction of the text window to hold column heading on the screen, while the bottom part of the screen scrolls.	
ANIMATION	15% (P)	Animation basically involves drawing a shape at a location, calculating a new location, 'undrawing' the shape, and redrawing the shape at the new location. Discuss the speed and simplicity of doing animation with sprite graphics and shape tables. Discuss the problems associated with animating against complex backgrounds (putting the background back to normal after the animated shape moves).	
COLOUR COMMANDS	5% (P)	Commands will be very specific to the system being used.	
Clark & LaBarre – Chapter 5 (151) – Chapter 13 (377-380, 390-391)		If your systems have high and low resolution graphics, emphasize the difference in the commands used to generate colour in each.	
SOUND COMMANDS	5% (P)	Stress the use of sound to attract attention in unusual circumstances, for example, when 'dangerous' routines are entered like deletion routines, and when keyboard entry errors are detected. Mention generation of sound by 1) poking the speaker address, 2) using special commands that specify frequency(ies) duration and volume, and 3) using Key Commands (like Control G, for example).	

MODULE 17: SYSTEMS ANALYSIS AND PROGRAM DEVELOPMENT

Status: Elective
Prerequisite: Module 1

Purpose

In this module[1] students systematically analyze and develop an appropriate algorithm to solve a specific problem or problems using a systems approach. They flowchart, code, and debug the program(s) that will achieve this goal.

Planning

For some students, this may be a first encounter with the formalized use of systems analysis in problem-solving situations. Therefore, a brief overview of a systems approach is recommended to provide an appropriate framework for actual analysis of an identified problem. Explain that a systems approach simply means that a series of preplanned logical procedures or steps are executed in a meaningful sequence in order to solve a specific problem. Identify the various phases of a systems approach – systems investigation, systems analysis, systems design, systems implementation, and systems evaluation and maintenance.

Teachers have considerable latitude in choosing the format of this module. The following are some possible alternatives:

- Assign the same problem to all students and have them complete it individually.
- Have students work individually on different problems.
- Have small groups of students work as cohesive units to solve the same problem.
- Have groups work independently on different problems.
- Have the entire class operate as one large group, with individual students assuming various roles and working on assigned activities within the structure of the entire classroom setting.

Simple problems (such as inventory control, payroll, utilities billing statement, credit card billing statement, mathematics or science application, etc.) are suitable for the individual approach. For a more involved case study approach involving group activities, the problem(s) would normally be much more comprehensive. For example, group activities might be: converting an organization from a manual operation to a computerized operation; including the conversion of business applications such as payroll to electronic data processing; computerized cash registers that automatically record the sale and adjust the inventory; reservations for a commuter airline, train or bus, etc. (See Module 5, pages 27 and 28 of the *Computer Processing 10-20-30 Curriculum Guide* for additional topic suggestions.)

---

1. Systems analysis may be defined as applying a systematic approach to the study and solution of problems. This is accomplished by reducing the problems into successively smaller components for individual study and solution, and usually involves the use of computers.

The class organization determines the depth and breadth of the specific problem(s) assigned.

Students should learn about various types of charting techniques before they begin detailed planning, so that they understand relationships between tasks, data, and timelines. Charting techniques include PERT (program evaluation and review technique), CPM (critical path method), Gantt-type chart, a decision table or decision tree, and graphical or pictorial charts such as line charts or grid charts.

The teacher and the students will need to monitor progress carefully throughout the entire exercise.

Resources

Several systems analysis textbooks are on the market. These have a variety of reading levels and they focus on a number of areas. If published materials are not available or not suitable, teachers may wish to develop and use their own problem applications. Students may benefit from access to a specific and comprehensive case study and appropriate software while they are working on the specific problem(s).

Evaluation

The evaluation methods should be related to the specific approach chosen (individual vs. group).

Theoretical 40%	Practical 60%

COURSE: COMPUTER PROCESSING

MODULE 17 -- SYSTEMS ANALYSIS AND PROGRAM DEVELOPMENT

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
SYSTEMS PROBLEM SOLVING Shelly – Chapter 10 (10.0-10.35) Sanders – Chapter 3 (66-73) – Chapter 12 (305-332)	5% (T)	Identify systems and the relationship of subsystems. Emphasize the use of a systematic approach to problem solving Consider the systems' department role and function and impact on the organization, and identify personnel and positions within the department. You may wish to include an overview of the life cycle of systems analysis. (Depending upon text/author, life cycles may vary from 4 to 8 distinctive steps.) Use transparencies, systems flowcharts, bulletin board displays, magazine/newspaper articles, and/or organization charts to enhance the presentation.	1. Theory tests (40%) a) Objective – multiple choice – fill-in blank – matching b) Subjective – definition – short answer – paragraph 2. Practical "hands-on" tests (60%)
SYSTEM INVESTIGATION Shelly – Chapter 10 (10.8-10.9) (10.29-10.33) Sanders – Chapter 12 (308-310)	5% (T)	Discuss the problem identification/definition and feasibility study -- an attempt to determine, at minimum expense, whether a reasonable solution exists. Discuss what analysis and techniques are necessary to identify problem(s), as well as whether a solution can be found. Identify potential algorithms, illustrations and examples from business, industry, research, construction, etc.	
SYSTEM ANALYSIS Shelly – Chapter 10 (10.10-10.12) Sanders – Chapter 12 (310-314)	5% (T)	Discuss and graphically illustrate the development of alternative solutions to the identified problem(s). Identify potential algorithms from associated data elements -- existing documentation, data flow, input, processing, and output considerations. (IPO chart)	
SYSTEMS DESIGN Shelly – Chapter 10 (10.12-10.23) Sanders – Chapter 12 (315-332)	10% (T)	Discuss and illustrate how the system is designed/developed on the basis of the previous steps. Identify a specific solution, use appropriate planning and research techniques, and develop an algorithm consistent within existing parameters and constraints. Use examples that have been identified earlier.	
PROGRAMMING TECHNIQUES Shelly – Chapter 11 (11.0-11.33) Sanders – Chapter 18 (477-510) – Chapter 19 (511-541) – Chapter 20 (542-583) – Chapter 21 (584-624) Mandell – Chapter 8 (153-170) Clark & Drum – Chapters 1-11 Clark & LaBarre – Chapters 1-8	60% (P)	Develop a flowchart. Code, debug, and run the program to solve the identified problem, or incorporate modifications to existing application software.	

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
PROGRAM DOCUMENTATION Shelly – Chapter 10 (10.26) – Chapter 12 (12.24-12.25) Sanders – Chapter 18 (500) – Chapter 19 (535-536) Mandell – Chapter 8 (155-158)	5% (T)	Documentation could include each step of the systems analysis (investigation through evaluation stage), as well as the actual computer program itself (input data, processing, and output information). Discuss and provide actual examples of appropriate documentation.	This module could be handled as a type of simulation exercise. It provides considerable opportunity for students to demonstrate their initiative and creativity, and their skills in planning, research, programming and communication. If this module is conducted as a simulation project, each phase of the project could be graded on a theory or practical basis, depending upon the approach taken. Alternatively, the entire module could be handled as an integrated project, with 100% of the evaluation based on a practical application. Suggested criteria: a) degree of difficulty of project b) originality and creativity c) efficiency of program d) output format e) documentation f) other factors Marks awarded should reflect: – objectives of the module – student involvement in the finished project
SYSTEMS IMPLEMENTATION Shelly – Chapter 10 (10.27) Sanders – Chapter 12 (322-324) – Chapter 15 (403-414)	5% (T)	Put the identified changes/modifications into place – – the computer program(s); hardware, if necessary; operation procedures/training; security; and formal testing procedures. Discuss the steps and procedures that are involved in this phase as well as the role of the systems analyst. Present the advantages and disadvantages of various methods of conversion.	
SYSTEMS EVALUATION AND MAINTENANCE Shelly – Chapter 10 (10.28) Sanders – Chapter 3 (73) – Chapter 12 (324) – Chapter 13 (338-342) – Chapter 13 (345-346)	5% (T)	Discuss terms as they are encountered. Basic terms should include at least: system, subsystem, systems investigation, systems analysis, systems design, systems implementation, systems documentation, systems evaluation, feedback, audit trail, program testing, systems flowchart, systems analyst, systems department. Discuss the need for feedback (to see if the changes are doing what was expected, and to make sure the system functions at an acceptable level). Identify the diminishing active role of the systems analyst after the new system is brought on stream.	

MODULE 18: MACHINE/ASSEMBLY LANGUAGE

Status: Elective
Prerequisite: Module 1

Purpose

This module is designed for students who have an interest in getting the most from their computer system through the use of low-level languages. The module also allows students to gain an appreciation for the advantages and disadvantages of a low-level language. The emphasis should be on the students' appreciation of the speed and versatility of a low-level language, and the trade-off that is made in its use (more difficult to program and debug than higher-level languages).

Planning

This module can be taught without the use of an assembler/editor, but students should spend only a small amount of time on this. Direct machine language programming should be reserved for minor patches to existing programs, and to teach appreciation of the relationship between machine and assembly language programming.

Students will not have time to become proficient at machine language or assembly language programming; rather, they should learn to appreciate the advantages and disadvantages of using low-level languages.

Resources

Specific manuals for the assembler/editor should be available for student use.

Evaluation

Theoretical 20%	Practical 80%

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
COMPARISON OF LANGUAGE Sanders – Chapter 19 (517-533) Mandell – Chapter 1 (16-20)	5% (T)	Place low-level languages in perspective with other components of computer systems. (See Appendix C.)	
HARDWARE CONFIGURATION	10% (T/P)	Some memory addresses are reserved by the nature of the hardware; e.g., I/O addresses, stack, etc.	
MEMORY MAP	10% (T/P)	Explain that the "system" uses various parts of the computer's memory to keep track of what the computer is doing. Normally, high-level languages are designed to respect these "reserved areas", but generally assembler and machine languages do not. Appendices F-1, F-2, and F-3 contain memory maps for a number of computers. If you are using an Apple computer and Applesoft BASIC, run the following program. It demonstrates the importance of the advanced programmer understanding the memory configuration of the machine being used. The program results in garbage being stored in the elements of the array that are created in the high resolution area:	

```
 10   REM LOMEM: 16384 WILL MOVE THE BEGINNING OF PROGRAM TO BEYOND HI-
      RES PAGES AND PREVENT COLLISION
 20   DIM X(3000):  REM CREATE A LARGE ARRAY
 30   REM DRAW AN IMAGE IN HIGH RES GRAPHICS
 40   HGR
 50   HCOLOR = 3:  HPLOT 10, 10 TO 10, 100
 60   HPLOT TO 100,100
 70   HPLOT TO 100,10
 80   HPLOT TO 10,10
 90   REM NOW PUT VALUES INTO THE ARRAY AND SHORTLY GARBAGE WILL APPEAR
      ON SCREEN
100   FOR I = 600 TO 3000
110   : : PRINT I; "  ";
120   : : LET X(I) = I
130   NEXT I
```

TOPIC/RESOURCE CORRELATION	TIME	TEACHING SUGGESTIONS	EVALUATION APPROACHES
MEMORY MAP (cont'd)		Discuss the effects of commands that change the computer's utilization of memory (e.g., HIMEM, LOMEM, MAXFILES).	
STORAGE LOCATION	10% (P)	Relate the named storage locations used in the assembler and high-level languages with the numeric allocation of storage locations.	
ASSEMBLY LANGUAGE SYNTAX	15% (P)	Have students write and run an application subroutine to enhance the features of an existing program.	
PROGRAM MODIFICATION	15% (p)	Check the program for problems, accuracy and applicability and make appropriate improvements.	
PROGRAM APPLICATIONS	35% (p)	Identify applications that are particularly suitable for machine/assembly languages. Have students develop a program in assembly language.	

EVALUATION

EVALUATION

Evaluation of student progress should be based on the objectives of the program and of the modules included in the course. Computer processing is primarily a practical "hands-on" course, but each module includes THEORETICAL and PRACTICAL components. Assessment should include both components, with the marks allocated reflecting the amount of time spent on each component. In addition, the evaluation of practical topics should involve practical activities.

The "Teaching Suggestions" include recommendations for evaluating the THEORETICAL and PRACTICAL components of each module. The following chart indicates the proportions for each module:

Module	Theoretical	Practical
1	100	
2	25	75
3	25	75
4	25	75
5	30	70
6	25	75
7	25	75
8	20	80
9	25	75
10	20	80
11	20	80
12	15	85
13	35	65
14	25	75
15	15	85
16	35	65
17	40	60
18	20	80

Specific suggestions for evaluating programming assignments are provided in the following sections.

Overview

Guiding principles for evaluating students' programs should be:

1. Clarity of Computer Code
    - modularity of program design (one segment of code doing one job)
    - use of descriptive variable names
    - generous inclusion of comments in listings to identify the program, modules within the program, algorithms used, and variable names
    - restricted use, or clear explanation, of any "tricky" code; i.e., a code that is accurate and even efficient but not obvious in its method of operation.
2. Modifiability of the Program
    - use of a variable, set near the beginning of a program, to store a value which is used throughout a program (instead of the actual value). If the value changes, only one change is required in the program.
3. "User-Friendliness"
    - screen displays that do not scroll
    - input error checks with helpful error messages
    - clean, clear, uncluttered and professional-looking screen and/or printer displays.

These principles should be weighted more heavily as students gain more experience.

Marking Guide

The following seven points can be used as a guide for marking students' programming assignments. The weighting of the various components will change depending upon the purpose of the program and the experience of the students. In Appendix F, suggested weightings are given for each of the seven items wherever programs are to be used in evaluation.

1. Program Description

   - A narrative of the purpose of the program, the inputs required, and the output given.

2. Symbol Table

   - A table showing every variable used in the program along with an explanation of what it is used for. (This can be done on a separate sheet or in comments at the beginning of the program.)
   - List of descriptive variable names.

3. Documentation of the Algorithm Used in the Program

   - A flowchart; or a structured diagram, hierarchy chart, Warnier-Orr diagram, or pseudocode; or any combination of these. (It is very helpful when marking to have students indicate on the flowchart the appropriate line numbers that perform each executable statement.)

4. Program Listing

   - Use of comments, program structure and effectiveness should be noted.
   - Program should follow the documentation in step 3 above.

5. Test Run

   - If the students hand in a test run, all test data should be shown – – marginally acceptable, marginally unacceptable and typical values.

6. Ease of Use "Userability"

   - Screen scrolling, validity checks with helpful error messages, proper spelling and format, brief introduction regarding the purpose of the program.

7. Presentation

   - Neatness and orderliness of documentation, use of flowchart template.

RESOURCES

BASIC TEXTS

Appleworks. Computer software for word processing, spreadsheet and data base.
(Modules 3, 4, 5, 7) Apple Computer Inc., 1983.

A.S.A.D., A Case Study. (Module 17) Oxford University Press, 1984. Teacher's Guide.

Brady. Apple Graphics Activities Handbook. (Modules 10, 16)
Prentice-Hall Canada Inc., 1984.

Braun, P. Computer Projects. APPLE. (Modules 8-11) Vancouver, B.C.:
Pacific Learning Institute, 1983.

Clark, J.F., and J.E. LaBarre. Advanced Structured BASIC. Cincinnati, Ohio:
South-Western Publishing Co., 1985.
Study Guides and Problems, Teacher's Manual.

Clark, J.F., and W.O. Drum. Structured BASIC. 2nd Edition. Cincinnati, Ohio:
South-Western Publishing Co., 1983.
Study Guides and Problems, Teacher's Manual.

Mandell, S.L., and C.J. Mandell. Computer Science with Pascal for Advanced Placement Students.
St. Paul, Mn,: West Publishing Company, 1985.
Teacher's Manual with Test Bank.

McRitchie, M.M. Programming in BASIC: A Complete Course. (Modules 8-10)
Toronto: Holt, Rinehart and Winston of Canada Ltd., 1982.
Solutions Manual.

Sanders, D.H. Computers Today. 2nd Edition. U.S.A.: McGraw-Hill Ryerson Limited, 1985.
Student Study Guide and Instructor's Manual.

Shelly, G.B., and T.J. Cashman. Introduction to Computers and Data Processing.
(Modules 1, 8-10, 17) Fullerton, C.A.: Anaheim Publishing Company, 1980.
Student Workbook and Student Guide.

LEARNING RESOURCES

The following software is available from the Learning Resources Distributing Centre as of June 1986.

S/W – Word Processing

1. Apple Writer II-e by Paul Lutus (1983, Cost: US $195.00).

   Status: Recommended.

   Apple II-e only, 64K, 1 disk drive, monochrome monitor (40 or 80 line), printer.

   Evaluation: Document creation is straightforward and text movement is easy once the process is learned. Search and replace feature is efficient. Error messages lack clarity. File loading, storing and deleting facilities are efficient. No password security available.

   Correlation to curriculum:

2. Bank Street Writer by International Educations (1982, Cost: US $125.00).

   Status: Recommended.

   Apple format only, requires Apple II+/Apple II-e, 48K, 1 disk drive, monitor (printer recommended).

   Simple documents can be prepared easily. Scrolling, uses Write and Edit modes. Deletion process is time-consuming as process is by characters only.

   Correlation to curriculum:

3. Format II (enhanced) by M.A.R. Hardwick and G.K. Bechman (1983, Cost: US $150.00).

   Status: Recommended.

   Equipment: Apple II family, 48K, 1 disk drive, monochrome monitor, shift key adapter for Apple II+ ($5.00).

   A document is created by selecting "New Page" from menu. Editing can be done by one of two modes: "Enter Text" and "Format Text". Easy to use. Search and replace function is excellent. Storing process is easy to remember. No password security is available.

   Correlation to curriculum:

4. Screenwriter II published by Sierra, 36575 Mudge Ranch Road, Coarsegold CA 93614. (1983, Cost: US $130.00).

   Status: Recommended.

   Equipment: Apple II + /II-e, 48K, disk drive, monitor and printer. Apple format only.

   Document creation is straightforward. Command structure is logical. Variety of scrolling options. Editing operations are efficient in 40 and 80 character modes; slow in 70 column mode.

   Correlation to curriculum:

5. Touchtype by Harvey R. Brown, Software Spirit, Box 825, Spirit River, Alberta T0H 3G0 (1984, Cost: C $100.00).

   Status: Recommended.

   Components: 1 disk, 1 back up, reproducible user's guide (55pp).

   Requirements: Apple II family, 48K, 1 disk drive, mono monitor, shift key modifications, 80 column card, Apple format only.

   Description: Document creation is straightforward. Switching from one file to another in memory is easy. Ability to load files by number is a timesaver. No password security is available.

   Correlation to curriculum:

6. Simulated Computer by Edusoft, Berkeley, CA, Steketee Educational Software (1982, US $30.00).

   Status: Recommended.

   Available from Alphatel Systems, Edmonton.

   Topic: Computer Literacy.

   Components: 1 disk, user's guide (13pp).

   Requirements: Apple II + /II-e, 48K, 1 disk drive, monitor Atari and TRS 80 formats.

   Content: Evaluation (an introductory program).
   Student types into computer memory the instruction steps outlined in the program and the user's guide, observes the results on the computer, and answers questions posed in the guide. Does not teach how to write machine language.

   Correlation to curriculum:

APPENDICES

SAMPLE DOS EXERCISE FOR AN APPLE LAB

For each student, make up one diskette that contains at least one of each of the following types of files:

- Applesoft program file (e.g., a lores graphics program)
- Integer BASIC program file
- Text file
- Binary file (e.g., a hires graphics picture)

Some of these files should be locked. Before the material is covered in detail, have the students CATALOG, LOAD, SAVE, LOCK, UNLOCK, RUN and DELETE these files, paying attention to the "disk in use" light to determine whether the command is a DOS command. Have students record what happens for each procedure.

Suggested Sequence of Commands

1. CATALOG
2. RUN <program name>
   This can be a simple text program that requires no input from the user and leaves information on the screen.
3. HOME
4. RUN <program name2>
   This can be a simple animated graphics program that leaves a picture on the screen.
5. TEXT
6. HOME
7. LOAD <program name>
8. LIST
9. NEW
10. LIST
11. CATALOG
12. UNLOCK <a locked program name>
13. CATALOG
14. LOCK <the program unlocked above>
15. CATALOG
16. DELETE <a locked file>
17. UNLOCK <a locked file>
18. DELETE <the file>
19. CATALOG
20. RUN <the deleted file name>
21. Get another disk with the deleted file still on it and LOAD <the file that was deleted off original disk>
22. Switch back to the original disk and SAVE <the file that was deleted>
23. CATALOG the original disk
24. LOCK <the file just saved>

PROFESSIONAL DOCUMENTATION STANDARDS FOR PROGRAMS

Professional Expectations

Creating computer algorithms and programs is a vital step in the study of computer processing. Techniques of structured problem-solving and programming should be emphasized from the start––even before learning the instruction of a specific language. The concept of "good programming", rather than "good coding", should be stressed. A program may be "coded" correctly and concisely in a particular language, but be poorly structured and hard to read. This kind of "coding" is to be avoided.

Documentation Standards

Documentation can be defined as "the collection of written information about a computer program". Some of this information is written on paper, and some in the program listing itself.

There appears to be as many different documentation standards for programs as there are textbooks. The following combination of several standards works well in a high school setting.

1. The Program Description should be a statement of what the program is to do. The description should also include a list of the values to be input, and the values that the program will output.

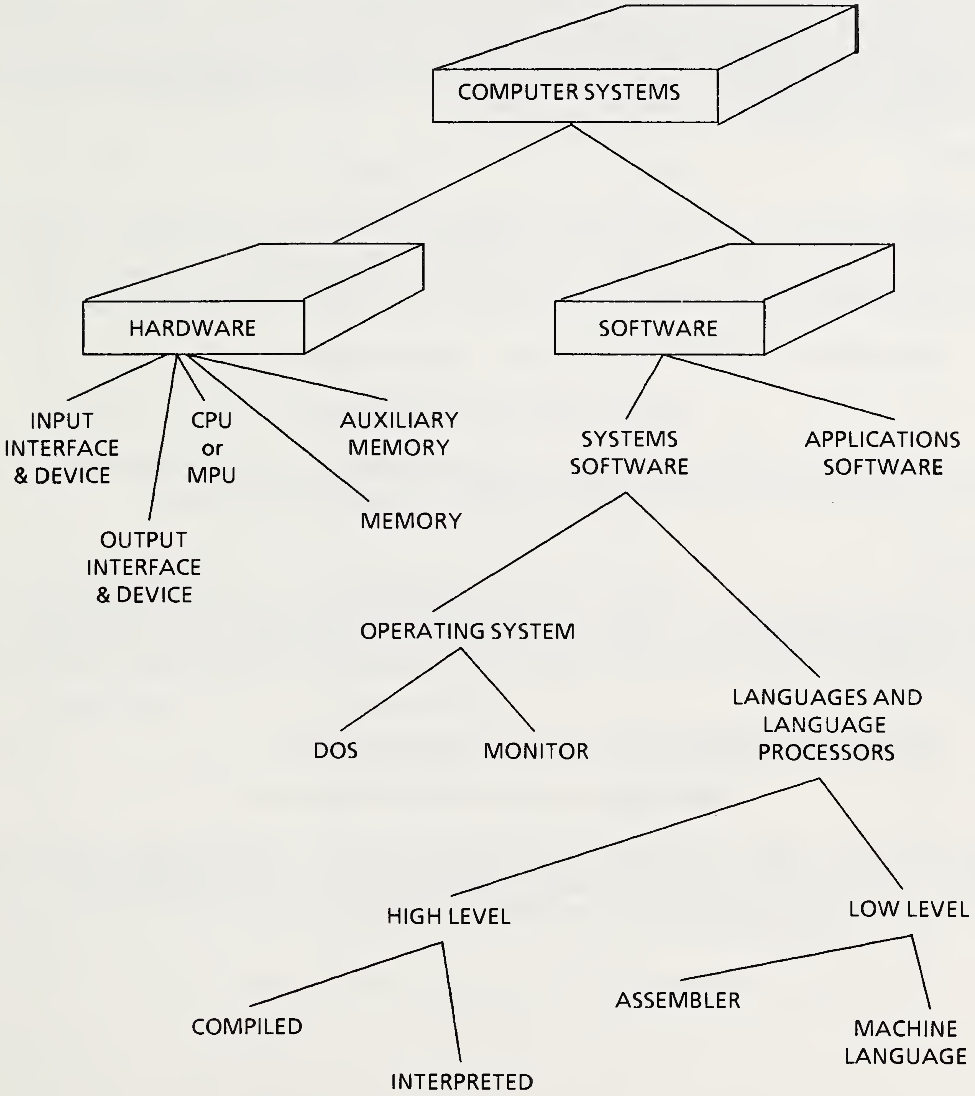
2. The Symbol Table, which can be in comment (REM) statements at the beginning of the program or on paper, is a listing of all variable names used, their data type (real, integer, string), and what they are used for in the program.

3. There should be some form of documentation regarding the algorithm used in the program. This is commonly a flowchart, but it could also be a structure diagram, hierarchy chart, or pseudocode.

4. Program Listing should include appropriately embedded comment statements or blank lines to help identify program segments or blocks of instructions, the use of indentation of nested loop instructions, and error rapping instructions if appropriate.

5. A sample test run of the program can be a hard copy printout. Evaluation may be done on the printout handed in, or the instructor can have diskettes handed in and run the programs for evaluation.

COMPUTER SYSTEM RELATIONSHIPS



SAMPLE PROGRAMMING PROJECT AT TWO LEVELS OF DIFFICULTY

The first version of this programming assignment, which every student will complete, involves the relatively simple use of IF/THEN for looping and decision-making. Completion of this first version will be worth 80% of the mark. The second version, which will be done only AFTER the first version is completed, involves the more complex use of IF/THEN. The second version is provided to challenge students who can handle more complex programs.

Level 1

In standard format, write a BASIC language program that will allow the operator to enter the sex (using M or F) and average mark (between 0 and 100%) of ten students. For each student, your program should print out BOY or GIRL, and HONOURS or NOT HONOURS, in sentence format. Do not use the FOR/NEXT statement to implement the ten repetitions; use IF/THEN and counters.

For input of "M" for sex and "85" for mark, the printout will be:

```
THIS IS A MALE HONOURS STUDENT.
```

After completing the ENTIRE first version including all test runs, etc., ask for the second version. This version is a modification of the first version and adds some more complex features.

Level 2

Modify the first version of the program so that the operator may enter the number of students from the keyboard, and so that the number may be between one and twenty. Have the program print out, in sentence format, the sex of the student (BOY or GIRL) and the letter grade for the student's mark.

For input of "M" for sex and "85" for mark, the printout will be:

```
THIS IS A MALE STUDENT WITH LETTER GRADE A.
```

When the total number of students has been entered, print out a class average of the students' marks. Ask the operator if he or she would like to do the average for another class. If so, the printout asks for number of students in the class. This program is to be in standard format as well.

COMPUTING LANGUAGE PRIMITIVES AS IMPLEMENTED IN BASIC AND PASCAL

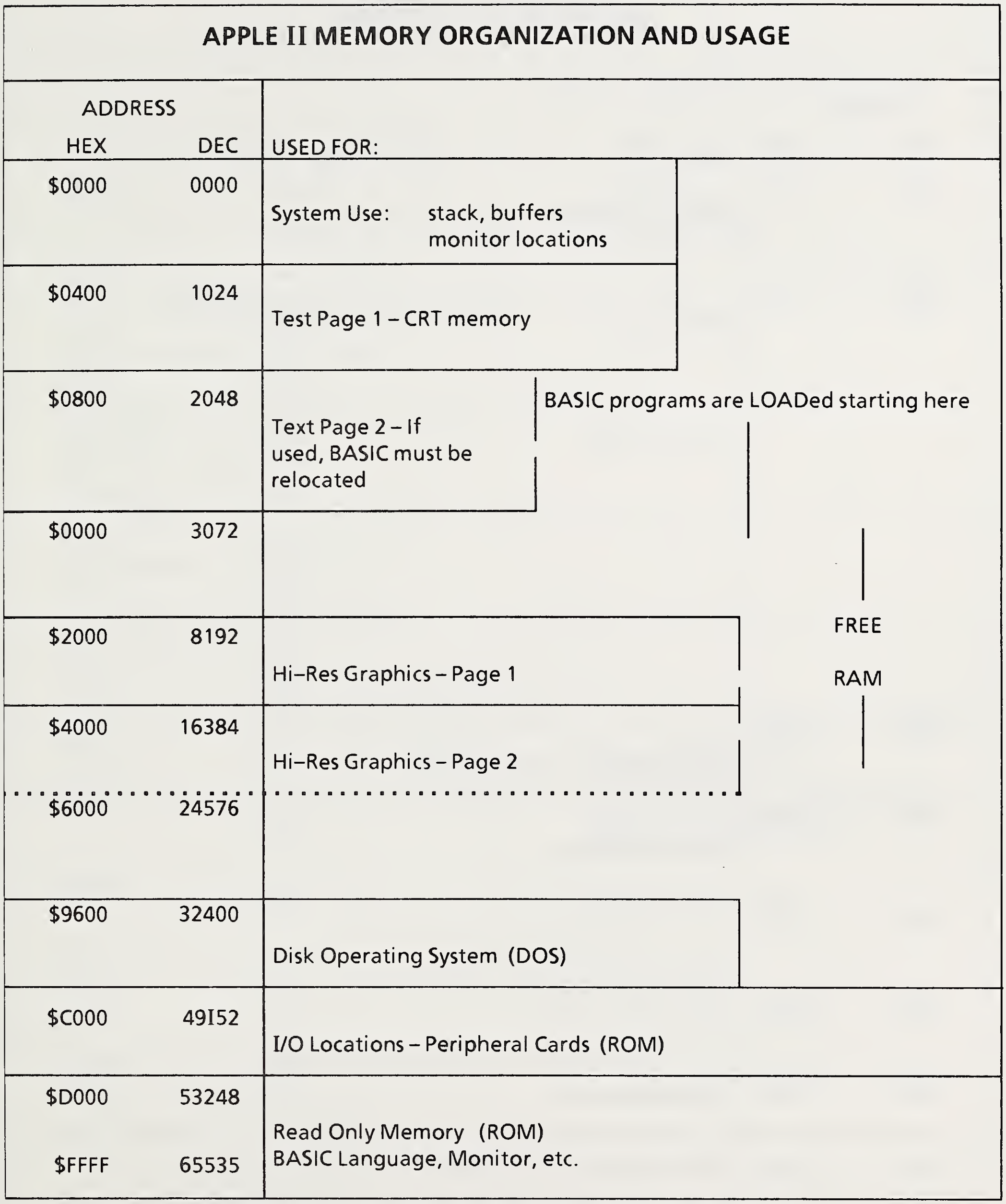
	LANGUAGE SPECIFIC COMMAND	
PRIMITIVE	PASCAL	BASIC
INPUT	READ	INPUT
	READLN	
	GET	
		GET
		READ/DATA
OUTPUT	WRITE	PRINT
	WRITELN	
	PUT	
ASSIGNMENT	Assignment operator (: =)	LET
REPETITION	REPEAT/UNTIL	
	FOR/DO	FOR/NEXT
	WHILE/DO	IF/THEN's
DECISION/SELECTION	IF/THEN/ELSE	IF/THEN/ELSE
	CASE	ON GOSUB/GOTO

APPENDIX F

SUGGESTED EVALUATION OF COMPUTER PROGRAMMING ASSIGNMENTS

	MODULE										
Marking Guide Item	8	9	10	11	12	13	14	15	16	17	18
1. Program Description	5	5	5	5	10	5	5	10	5	20	5
2. Symbol Table	5	5	5	5	5	5	5	5	5	10	5
3. Algorithm Documentation	10	15	20	25	30	25	30	30	30	30	25
4. Program Listing	25	25	20	20	15	30	20	15	20	10	20
5. Test Run	35	30	30	30	25	20	25	25	25	20	30
6. Ease of Use	10	10	10	10	10	10	10	10	10	5	10
7. Presentation	10	10	10	5	5	5	5	5	5	5	5
	100	100	100	100	100	100	100	100	100	100	100

APPLE II MEMORY ORGANIZATION AND USAGE



ADDRESS HEX	ADDRESS DEC	USED FOR:	
$0000	0000	System Use: stack, buffers monitor locations	
$0400	1024	Test Page 1 – CRT memory	
$0800	2048	Text Page 2 – If used, BASIC must be relocated	BASIC programs are LOADed starting here
$0000	3072		
$2000	8192	Hi–Res Graphics – Page 1	FREE RAM
$4000	16384	Hi–Res Graphics – Page 2	
$6000	24576		
$9600	32400	Disk Operating System (DOS)	
$C000	49152	I/O Locations – Peripheral Cards (ROM)	
$D000 $FFFF	53248 65535	Read Only Memory (ROM) BASIC Language, Monitor, etc.	

* Dotted lines represent memory areas used at the discretion of the user. Solid lines are for system use.

CBM 8032 MEMORY ORGANIZATION AND USAGE

ADDRESS		
HEX	DEC	USED FOR:
$0000	0000	Working Storage: stack, pointers, etc.
$0400	1024	Text Programs BASIC programs are LOADed here FREE RAM
$8000	32768	Screen Memory and I/O (RAM)
$9000	36864	Expansion ROM
$B000	45056	Start of BASIC 4.0
$E000	57344	Screen Editor (2K)
$E800	59392	I/O Memory (2K)
$F000	61440	Operating System (OS) (4K)

MEMORY ARCHITECTURE

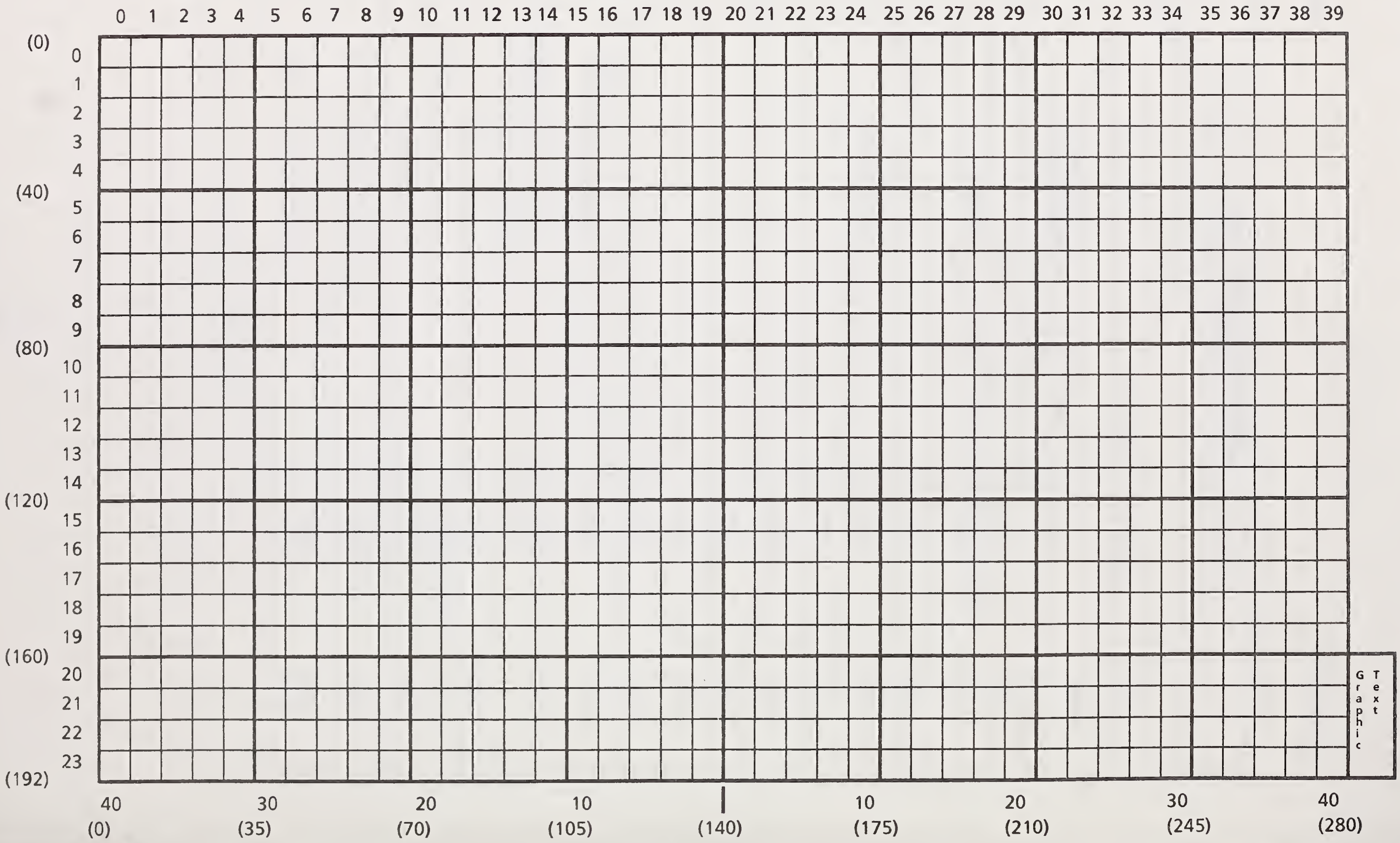
MEMORY ORGANIZATION AND USAGE

Hex	Decimal	Used For
00000	0K 32K	DOS Software Start of User Area
10000	64K	
20000	128K	
30000	192K	
40000	256K	User Memory
50000	320K	
60000	384K	
70000	448K	
80000	512K	
90000	576K	
A0000	640K	Reserved for Graphics
B0000	704K 736K	4K Monochrome Display Buffer 16K Colour Display Buffer
C0000	768K 800K	Fixed Disk ROM
D0000	832K	
E0000	896K	
F0000 FE000	960K 1016K	ROM Area

TEXT AND GRAPHIC FORMAT SHEET

DISPLAY LAYOUT FORM

Program ____________ **Author(s)** ________________ **Frame** ________________



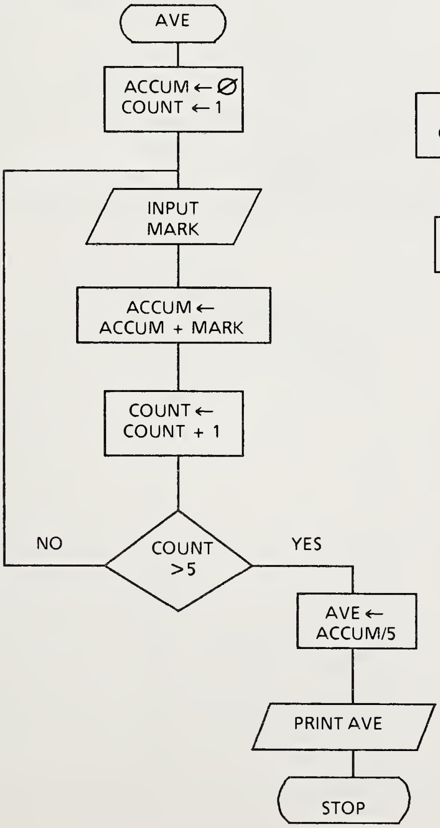
COMPARISON OF PSEUDOCODE, STRUCTURE DIAGRAM AND FLOWCHART FOR STATEMENT OF ALGORITHMS

These three different methods can be used to specify an algorithm to calculate the average of five numbers entered from the keyboard and print the result at the end.

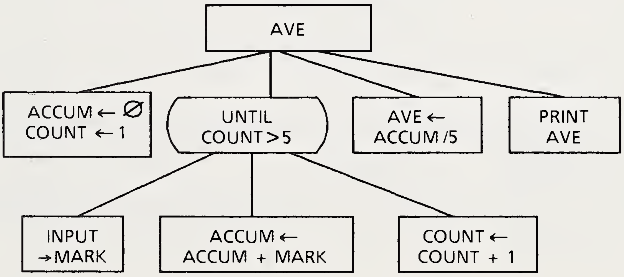
A. Pseudocode

```
1. Initialize
   a) set ACCUMULATOR to 0
   b) set COUNTER to 1
2. Input and accumulate marks
   a) repeat until COUNTER>5
        i.   input a mark to MARK
        ii.  add MARK to ACCUMULATOR
        iii. increment COUNTER
3. Calculate average
   a) set AVERAGE to ACCUMULATOR divided by 5
4. Print result
   a) print AVERAGE to screen
```

B. Flowchart



C. Structure Diagram



CORRELATION OF COURSE CONTENT WITH LEARNING RESOURCES
COMPUTER PROCESSING SUBJECT STRAND

COURSE CONTENT BY MODULE	INTRO. TO COMPUTERS & DATA PROCESSING (SHELLY & CASHMAN)	PACIFIC LEARNING INST. COMPUTER PROJECTS (PETER BRAUN)	PROGRAMMING IN BASIC A COMPLETE COURSE (MARGARET MCRITCHIE)	APPLE SOFTWARE
MODULE 1: Computer Information Systems	Chapters 1, 2, 8			
MODULE 2: Overview of Software	Chapters 1, 2			Selected packages (dependent on program focus, hardware, level of student expertise)
MODULE 3: Applications – Data Entry	Chapters 1, 3, 5, 6	Project 1		Selected packages (dependent on program focus, hardware, level of student expertise)
MODULE 4: Applications – Word Processing	Chapter 3, pp. 3.14-3.15 Chapters 5, 6			Touchtype Appleworks Applewriter II Bank Street Writer Format II (Enhanced version) Screenwriter I
MODULE 5: Applications – Computer Simulations	Chapter 10, pp. 10.7-10.27 Resources to be identified by Instructor			Selected packages (dependent on focus of simulation)
MODULE 6: Applications – Data Base	Chapter 8, pp. 8.1-8.35 Chapter 9, pp. 9.1-9.16			Appleworks
MODULE 7: Applications – Electronic Spread Sheet				Appleworks Visicalc (up-graded version)
MODULE 8: Overview of Introductory Programming Language	Chapters 1-3 (Selected Topics)	Projects 1, 2	Chapters 1-6 (Selected Topics)	
MODULE 9: Fundamentals of Input/Output		Projects 3-6		
a) Input/Output Processes	Chapter 5, pp. 5.1-5.35 Chapter 6, pp. 6.1-6.25		Chapter 3	
b) Processing of Data	Chapter 4, pp. 4.1-4.20		Chapter 1, pp. 4-6	
c) Arrays	Appendix A, pp. A.30-A.36		Chapters 7, 8	
d) Library and Operating System Functions	Appendix A, pp. A.29-A.41 Workbook Appendix A, pp. A.1-A.8		Chapter 9	
e) Programming Applications	Appendix A, pp. A.37, A.41, A.42 Workbook Appendix A, pp. A.9, A.10			
f) Data Communications	Chapter 8, pp. 8.1-8.33			
MODULE 10: Introduction to Advanced Computer Programming Techniques		Projects 7-10 (Selected Sections)		
a) Numbering Systems	Appendix C, pp. C.1-C.9		Chapter 12, pp. 585-588	
b) Professional Programming Techniques	Chapter 12			
c) Subroutines	Chapter 11		Chapter 11, pp. 508-524	
d) Multiple Dimension Arrays			Chapter 8	
e) Introduction to Graphics	User Manual Specific to Computer			

CORRELATION OF COURSE CONTENT WITH LEARNING RESOURCES

COMPUTER PROCESSING SUBJECT STRAND (CONTINUED)

MS DOS SOFTWARE	STRUCTURED BASIC (CLARK, DRUM)	ADVANCED STRUCTURED BASIC (CLARK, LABARRE)	COMPUTERS TODAY (SANDERS)	COMPUTER SCIENCE WITH PASCAL I (MANDELL, MANDELL)
			Chpaters 1, 2, 4, 5, 7 - 17	
Selected packages (dependent on program focus, hardware, level of student expertise)			Chapters 8, 11, 12, 13	Chapter 1, Appendix H
Selected packages (dependent on program focus, hardware, level of student expertise)			Chapters 2, 6, 7, 15	
Word Wordstar 2000 Plus WordPerfect			Chapters 2, 11, 14	
Selected packages (dependent on focus of simulation)			Chapter 14, pp. 367, 368 Chapter 16, pp. 445, 446	
Enable Learn Power Base Reflex			Chapters 2, 10, 12, 13, 14	
Twin 2			Chapter 4, p. 82 Chapter 8, pp. 219, 222, 223 Chapter 14, pp. 365, 366	
	Chapters 1-9 Appendix B, D		Chapters 20, 21	Chapters 1-8 Appendix H, I
	Chapters 10, 11, 12 Appendix B, D	Chapters 1-7 Appendix C	Chapters 5, 6, 7, 10, 18, 21	Chapters 1, 9, 10, 11, 13
		Chapters 1, 3, 4, 5, 6, 12, 13	Chapters 5, 21	Chapters 2, 8, 9, 10, 11

CORRELATION OF COURSE CONTENT WITH LEARNING RESOURCES
COMPUTER PROCESSING SUBJECT STRAND (CONTINUED)

COURSE CONTENT BY MODULE	INTRO. TO COMPUTERS & DATA PROCESSING (SHELLY & CASHMAN)	PACIFIC LEARNING INST. COMPUTER PROJECTS (PETER BRAUN)	PROGRAMMING IN BASIC A COMPLETE COURSE (MARGARET MCRITCHIE)	APPLE SOFTWARE
MODULE 11: Advanced Computer Programming Techniques				
a) Types of Files	Chapter 7, pp. 7.1-7.22	Projects 5-9 (Selected Sections)	–	
b) File Handling Procedures	User Manual Specific to Computer			
MODULE 12: Extended Programming Project	Chapter 12, pp. 12.1-12.33	Projects 7-10 (Selected Sections)	Chapter 1, pp. 6-9	
MODULE 13: Introduction to Second Programming Language	Chapter 12, pp. 12.1-12.6	–	–	References are specific to the programming language being taught
MODULE 14: Applications in Second Programming Language	–	–	–	
MODULE 15: Extended Project in Second Programming Language	Chapters 11, 12	Projects 4, 6, 7	Chapter 11, pp. 508-524 Chapter 8	
MODULE 16: Graphics	–	–	–	Graphics references are specific to the computer used
MODULE 17: Systems Analysis and Program Development	Chapters 10, 11, 12	Projects 4-10	Chapters 8, 9, 10, 11, 12	
MODULE 18: Machine/Assembly Language	Chapter 12	–	–	References are specific to the computer used

Notes:
1. The basic resources cover the majority of the topics.
2. The above correlations are suggested as guidelines to assist in teacher planning.
3. Additional computer software packages have been identified at the supplementary level. Refer to Clearinghouse Reports.

CORRELATION OF COURSE CONTENT WITH LEARNING RESOURCES
COMPUTER PROCESSING SUBJECT STRAND (CONTINUED)

MS DOS SOFTWARE	STRUCTURED BASIC (CLARK, DRUM)	ADVANCED STRUCTURED BASIC (CLARK, LABARRE)	COMPUTERS TODAY (SANDERS)	COMPUTER SCIENCE WITH PASCAL (MANDELL, MANDELL)
	Chapters 13, 14	Chapters 4, 9, 10, 11	Chapter 7	Chapters 12, 14, 15
		Chapter 1, pp. 2-41	Chapter 18, pp. 479-504 Chapter 19, pp. 512-536	
	Chapters 1-11 Appendices Section	Chapters 1-8 Appendices Section	Chapter 19, pp. 517-532 Chapter 20, pp 542-583 Chapter 21, pp. 584-624	Chapter 1, pp. 16-20 Chapters 2-11, pp. 25-232 Appendices Section
		(Same references as in Module 13)		
		Chapter 5, pp. 145-169 Chapter 12 Chapter 13	Chapter 7, pp. 195-198	
	Chapters 1-11	Chapters 1-8	Chapter 3, 12, 13, 15, 18, 19, 20, 21	Chapter 8
			Chapter 19, pp. 517-533	Chapter 1, pp. 16-20

N.L.C. - B.N.C.
3 3286 07417012 3